UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

JÉFERSON CAMPOS NOBRE

Manutenção da Consistência do Estado dos Dados de Gerenciamento em Sistemas de Gerenciamento Autonômico baseados em Infraestruturas *Peer-to-Peer*

Dissertação apresentada como requisito parcial
para a obtenção do grau de
Mestre em Ciência da Computação

Prof. Dr. Lisandro Zambenedetti Granville
Orientador

Porto Alegre, Junho de 2010

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Jéferson Campos Nobre,

Manutenção da Consistência do Estado dos Dados de Gerenciamento em Sistemas de Gerenciamento Autonômico baseados em Infraestruturas *Peer-to-Peer* /

Jéferson Campos Nobre. – Porto Alegre: PPGC da UFRGS, 2010.

90 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR–RS, 2010. Orientador: Lisandro Zambenedetti Granville.

1. Gerenciamento de Redes. 2. Computação Autonômica. 3. P2P. 4. MAS. 5. TMS. I. Granville, Lisandro Zambenedetti. II. Título.

Para minha mãe Maria,
um exemplo de força, coragem e caráter.

AGRADECIMENTOS

À minha Mãe Maria, por todo o apoio aos meus estudos, pelos inúmeros ensinamentos durante toda vida e pela presença em todos os momentos difíceis;

À minha Namorada Maíra, por todo o amor, carinho, apoio, compreensão e também pela paciência nos momentos em que estive ausente;

Ao meu Orientador, Professor Lisandro Zambenedetti Granville, por toda a colaboração, compreensão nos momentos complicados e valiosas lições que contribuiram para o meu crescimento pessoal e acadêmico;

Aos Professores do Grupo de Redes de Computadores da UFRGS, Luciano Paschoal, Marinho Barcelos, Juergen Rochol, Maria Janilce Almeida, João Netto e Liane Tarouco, pelos ensinamentos;

A todos colegas do Grupo de Redes de Computadores da UFRGS, por todo o companheirismo, apoio e por todas as experiências que pudemos partilhar juntos durante esta caminhada.

“Ignorance more frequently begets confidence than does knowledge:
it is those who know little, not those who know much,
who so positively assert that this or that problem will never be solved by science”
– CHARLES DARWIN

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

AC	*Autonomic Computing*
AIS	*Alarm Indication Message*
AMD	*Autonomic Management Domain*
AME	*Autonomic Management Element*
ANM	*Autonomic Network Management*
API	*Application Programming Interface*
CMIP	*Common Management Information Protocol*
ECA	*Event-Condition-Action*
EVC	*Ethernet Virtual Connection*
ISO	*International Organization for Standardization*
IETF	*Internet Engineering Task Force*
IT	*Information Technology*
KB	*Knowledge Base*
LCK	*Locked Signal Message*
MAS	*Multi-Agent System*
MbD	*Management by Delegation*
MEF	*Metro Ethernet Forum*
MLM	*Middle Level Manager*
OAM	*Operations, Administration, and Maintenance*
P2P	*Peer-to-Peer*
PBNM	*Policy-Based Network Management*
PDP	*Policy Decision Point*
PEP	*Policy Enforcement Point*
QoS	*Quality of Service*
SLA	*Service Level Agreement*
SNMP	*Simple Network Management Protocol*

SPoF	*Single Point of Failure*
TLM	*Top Level Manager*
TMS	*Truth Maintenance System*
TTS	*Trouble Ticket System*
XML	*eXtended Markup Language*
EVC	*Ethernet Virtual Connection*

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

O Gerenciamento Autonômico de Redes é uma visão que utiliza princípios da Computação Autonômica para o Gerenciamento de Redes. Além disso, algum grau de descentralização é necessário para habilitar capacidades autonômicas completas. Uma alternativa interessante de infraestrutura para essa união é a utilização de *overlays Peer-to-Peer* (P2P). No entanto, a consistência do estado dos dados de gerenciamento entre os *peers* é um desafio importante. Mecanismos tradicionais para manter a consistência desses estados são implementados por meio de centralização, o que desperdiça algumas propriedades desejáveis de abordagens P2P. Em contraste com esses mecanismos, é proposto um mecanismo distribuído, escalável e robusto para a manutenção da consistência do estado dos dados de gerenciamento pela introdução de funcionalidades de Manutenção da Verdade Multiagente. Além disso, são propostas estratégias de comunicação para prover suporte a essas funcionalidades. São apresentados também estudos de caso para ilustrar as possibilidades da proposta: o gerenciamento cooperativo de falhas em enlaces Ethernet em provedores de serviços e a ativação distribuída de políticas de gerenciamento de redes. Experimentos simulados são realizados a fim de verificar as propriedades de escalabilidade e robustez da presente proposta.

Palavras-chave: Gerenciamento de Redes, Computação Autonômica, P2P, MAS, TMS.

ABSTRACT

Autonomic network management is a vision that brings Autonomic Computing principles to Network Management. Besides, it is necessary some level of decentralization to enable broad autonomic capabilities. An interesting alternative of infrastructure for this union is the utilization of Peer-to-Peer (P2P) overlays. However, the consistency of state of management data among peers is an important challenge. Traditional mechanisms to maintain consistency of these states are supported by some centralization which wastes some desirable properties of P2P approach. In contrast to these mechanisms, a distributed, scalable and robust mechanism to maintain the consistency of state of management data is proposed through the introduction of Multi-Agent Truth Maintenance features. Besides, communication strategies are proposed to support these features. Case studies are presented to show possibilities of this proposal: cooperative fault management of Ethernet links in service providers and distributed activation of network management policies. Simulated experiments are performed to verify the scalability and robustness properties of this proposal.

Keywords: Network Management, Autonomic Computing, P2P, MAS, TMS.

1 INTRODUÇÃO

A complexidade crescente das redes de computadores exige soluções sofisticadas para gerenciar a infraestrutura básica de comunicação e auxiliar os administradores de rede em suas tarefas cotidianas (KIND et al., 2008). A aplicação de conceitos da Computação Autonômica (*Autonomic Computing* - AC) em gerenciamento de redes, conhecida como Gerenciamento Autonômico de Redes (*Autonomic Network Management* - ANM), foi proposta como uma nova forma de solucionar algumas demandas enfrentadas pelo gerenciamento de redes tradicional, tais como a tolerância de graus de incerteza no plano de gerenciamento e o controle de ambientes altamente dinâmicos (PRAS et al., 2007).

Os conceitos utilizados para a definição da AC tiveram forte conotação biológica, sendo que uma das primeiras analogias utilizadas foi o Sistema Nervoso Autônomo, o qual, por exemplo, controla o ritmo do coração e a temperatura do corpo, liberando a consciência de lidar com funções de baixo nível, no entanto, vitais (KEPHART; CHESS, 2003). Da mesma forma, pode-se imaginar um sistema de gerenciamento de redes em que as funções básicas sejam automatizadas e a intervenção humana seja em forma de instruções de alto nível, como definição de objetivos ou políticas.

Os sistemas ANM aumentam a eficiência dos administradores de redes em suas tarefas, diminuindo a necessidade de intervenções manuais. Esse aumento de eficiência é realizado por meio de automações e/ou otimizações de detalhes operacionais das tarefas de gerenciamento (*e.g.*, manipulação de falhas) (CHAPARADZA, 2009). Dessa forma, os esforços dos administradores podem ser concentrados nos níveis mais altos de abstração do gerenciamento de redes. O conceito principal utilizado em um sistema ANM é a possibilidade do próprio sistema de gerenciamento manter e ajustar aspectos de sua operação em função de mudanças no ambiente. Esse conceito é também conhecido como autogerenciamento (*self-management*), sendo o mesmo frequentemente entendido por meio de diferentes aspectos (*e.g.*, autoconfiguração) (KEPHART; CHESS, 2003).

Os sistemas ANM podem ser implementados mediante diferentes modelos de distribuição, desde modelos totalmente centralizados até modelos altamente descentralizados. Atualmente, não existem evidências que possibilitem correlacionar a qualidade das ações autonômicas com o modelo de distribuição adotado. Alguns autores, no entanto, acreditam que algum nível de descentralização é necessário para atingir um ANM mais adequado (JENNINGS et al., 2007). A introdução de pequenas entidades autogerenciáveis ao longo de uma infraestrutura gerenciada pode habilitar sistemas ANM efetivamente distribuídos. Neste cenário, o modelo típico dos sistemas ANM descentralizados é baseado em um grupo de *Elementos de Gerenciamento Autonômico* (*Autonomic Management Elements* - AME), que executam tarefas de gerenciamento e interagem para formar um *Domínio de Gerenciamento Autonômico* (*Autonomic Management Domain* - AMD). Múltiplos AMDs podem ser integrados para formar um *Ambiente de Gerenciamento Autonômico*

(*Autonomic Management Environment*) (JENNINGS et al., 2007).

Diferentes tecnologias podem ser empregadas como infraestrutura para sistemas ANM descentralizados. Uma possibilidade interessante é a utilização de *overlays Peer-to-Peer* (P2P), os quais incorporam suas características nos sistemas ANM, tais como o suporte à colaboração em tarefas de gerenciamento (*e.g.*, reconhecimento de falhas), robustez na conexão das entidades de gerenciamento, e balanceamento de carga nas tarefas de gerenciamento (GRANVILLE et al., 2005). Existem algumas iniciativas investigando ANM baseado em P2P (KAMIENSKI et al., 2008) (FALLON et al., 2007) (KONSTANTINOU; YEMINI, 2009) (MARQUEZAN et al., 2008), e nessas iniciativas *peers* possuem propriedades encontradas nos AMEs. Além disso, *peers* com características similares (*e.g.*, gerenciando o mesmo dispositivo) podem ser organizados em grupos. Esses grupos de *peers* possuem propriedades encontradas em AMDs.

Informações de gerenciamento necessitam ser compartilhadas pelos *peers* (que formam o sistema ANM) para a realização das tarefas de gerenciamento de forma coordenada e coerente. Podemos citar, por exemplo, a utilização de *políticas*, as quais possuem um papel central em sistemas ANM (EMANICS, 2010) (KONSTANTINOU; YEMINI, 2009), podendo ser utilizadas para automatizar o processo de tomada de decisão no sistema. Assim, em um sistema ANM baseado em P2P e habilitado para políticas, os *peers* devem possibilitar a utilização das mesmas em decisões distribuídas de gerenciamento. É importante ressaltar a necessidade de que essas decisões sejam realizadas de forma consistente entre *peers*.

A utilização de *overlays* P2P como infraestrutura de sistemas ANM, no entanto, não oferece suporte aos requisitos de consistência dos estados das informações distribuídas na execução das tarefas de gerenciamento. Esta inconsistência pode ser causada por falhas na rede de comunicação (*e.g.*, perdas na troca de mensagens entre os *peers*) e recursos computacionais (*e.g.*, operação incorreta de *peers*). Além disso, inconsistências no estado das informações de gerenciamento podem ocorrer mesmo na operação normal de um sistema ANM baseado em P2P, devido à falta de sincronização entre os *peers*. Essa sincronização de informações é um desafio em um sistema distribuído assíncrono não confiável (FISCHER; LYNCH; PATERSON, 1985), como, por exemplo, um *overlay* P2P não estruturado.

Em um sistema ANM baseado em P2P, elementos gerenciados (*e.g.*, roteadores) podem ser controlados por um ou mais *peers*. Essa multiplicidade de *peers* pode ser utilizada para uma maior robustez no gerenciamento, por exemplo. Nesse caso, o estado das informações do elemento gerenciado se torna uma informação distribuída e replicada entre os *peers* que realizam seu gerenciamento. Inconsistências nessas informações podem levar a uma operação incorreta do sistema ANM. Por exemplo, o status de um enlace de um roteador (*e.g.*, interfaces Ethernet) pode ser divulgado diferentemente pelos *peers* controladores para elementos externos que requisitem essa informação (*e.g.*, estações de gerenciamento).

Os mecanismos utilizados para prover suporte à consistência das informações de gerenciamento nos sistemas ANM baseados em P2P utilizam (em geral) ainda alguma forma de centralização, tais como repositórios externos (MARQUEZAN et al., 2008) ou *super peers* (KAMIENSKI et al., 2008) (FALLON et al., 2007), desperdiçando oportunidades de *overlays* P2P não estruturados (*e.g.*, escalabilidade). Essa centralização faz os sistemas citados apresentarem algumas características não desejadas de sistemas cliente-servidor (*e.g.*, pontos únicos de defeito), apesar de serem sistemas baseados em P2P. Em outras abordagens, o mecanismo de distribuição das informações de gerenciamento não é clara-

mente definido (JENNINGS et al., 2007). Uma distribuição efetiva das informações de gerenciamento demanda um mecanismo que aprimore sua consistência, evitando incoerências. Além disso, a estratégia de comunicação do mecanismo precisa ser escalável e robusta.

Considerando a área de gerenciamento de redes, a introdução de sistemas ANM baseados em P2P não é uma nova área de pesquisa. No entanto, até agora, não existem investigações focadas nos desafios relacionados à consistência do estado das informações de gerenciamento utilizadas nesses sistemas. A introdução de *peers* como pequenas entidades autogerenciadas pode interferir na coordenação do sistema de gerenciamento como um todo, assim, são necessários mecanismos para o sistema manter boas características globais, como predição comportamental, desempenho aceitável e eficiência.

Neste trabalho são introduzidas funcionalidades de *Manutenção da Verdade Multiagente* (*Multi-Agent Truth Maintenance*) (HUHNS; BRIDGELAND, 1991), uma tecnologia oriunda da área de Sistemas Multiagente (*Multi-Agent Systems* - MAS), em um sistema ANM baseado em P2P. Os estados das informações de gerenciamento são organizados como um conjunto de crenças justificadas e compartilhadas pelos *peers*. Também são propostas estratégias de comunicação para possibilitar uma troca eficiente dessas crenças dentro de um grupo de *peers*. Essas estratégias são inspiradas em mecanismos relativos a processos biológicos observados na natureza (BABAOGLU et al., 2006), em especial, a *replicação*.

Nesse contexto, a maior contribuição do trabalho advém do fato de que a proposta não apenas apresenta bom desempenho na manutenção da consistência do estado das informações de gerenciamento, mas realiza essa manutenção de forma distribuída, escalável e robusta. Além disso, a proposta também permite explorar outras possibilidades em sistemas ANM baseados em P2P, tais como a integração de diferentes serviços de gerenciamento.

Esta dissertação é organizada da seguinte forma. No capítulo 2 é apresentada uma discussão sobre modelos de gerenciamento e também são abordados os trabalhos relacionados. No capítulo 3 é descrita a proposta e seus conceitos implícitos. No capítulo 4 é apresentada a avaliação da proposta. A dissertação se encerra no capítulo 5, no qual são apresentadas as conclusões e os trabalhos futuros

2 FUNDAMENTAÇÃO TEÓRICA E TRABALHOS RELACIONADOS

Neste capítulo serão apresentados alguns dos principais tópicos relacionados com os assuntos desta dissertação. Primeiramente serão apresentados modelos de gerenciamento de redes, especialmente o modelo de Gerenciamento Autonômico baseado em *Peer-to-Peer* (P2P). Em seguida, serão apresentados trabalhos relacionados à proposta descrita nesta dissertação.

2.1 Modelos de Gerenciamento de Redes

Não existe uma taxonomia amplamente aceita que defina e caracterize quais são os modelos de gerenciamentos de rede existentes. Alguns autores dividem a área de gerenciamento em centralizado, fracamente distribuído, fortemente distribuído e gerenciamento cooperativo, considerando, para tanto, o número de elementos gerenciados e a escalabilidade do sistema (SCHöNWALDER; QUITTEK; KAPPLER, 2000). Outros autores classificam o gerenciamento de redes de acordo com modelos organizacionais (MARTIN-FLATIN; ZNATY; HABAUX, 1999) (LEINWAND; CONDROY, 1996). Dessa forma, o presente trabalho não segue uma taxonomia específica para apresentar os modelos de gerenciamento, mas procura demonstrar aqueles que são aceitos como os mais importantes pela comunidade de gerenciamento de redes.

Os modelos a serem apresentados são: centralizado, por delegação, baseado em políticas e autonômico. Da mesma forma, serão apresentadas aplicações de tecnologia P2P no gerenciamento de redes e a ligação dessa aplicação com o modelo de gerenciamento autonômico. Além de serem efetivamente aceitos na comunidade de gerenciamento, esses modelos são particularmente importantes, pois possuem características relevantes para o desenvolvimento do presente trabalho.

2.1.1 Gerenciamento Centralizado

O modelo de gerenciamento de redes centralizado foi o primeiro a ser utilizado, principalmente devido à sua facilidade de implementação e arquitetura simplificada. Nesse modelo, existe uma única estação de gerenciamento de rede, responsável por concentrar todo o processamento das tarefas de gerenciamento e um conjunto de agentes, os quais estão dentro dos dispositivos gerenciados e cuja tarefa está limitada a coletar dados (MARTIN-FLATIN; ZNATY; HABAUX, 1999). A Figura 2.1 ilustra a estrutura do modelo de gerenciamento de redes centralizado. Como pode-se perceber pela Figura 2.1, os dispositivos gerenciados estão diretamente ligados à estação de gerenciamento.

A abordagem gerente-agente é intimamente relacionada com o modelo de gerencia-

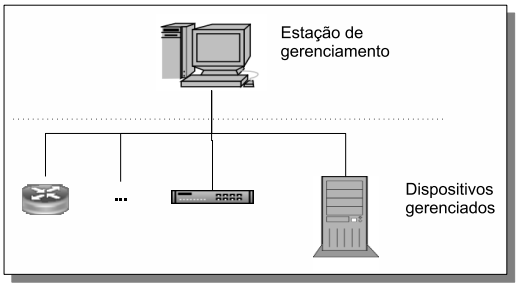

Figura 2.1: Modelo de gerenciamento centralizado

mento centralizado, em que apenas um gerente controla muitos agentes. O gerente está localizado na estação de gerenciamento, enquanto os agentes são entidades de software localizadas nos dispositivos gerenciados. Nessa abordagem, os gerentes acessam objetos gerenciados em dispositivos selecionados, onde o agente responde a essas requisições, além de enviar, assincronamente, informações sobre o dispositivo gerenciado. Dessa forma, o gerente concentra os alertas e eventos da rede, as informações de gerenciamento e o acesso às aplicações de gerenciamento. O *framework* de gerenciamento *de facto* para redes TCP/IP, *Simple Network Management Protocol* (SNMP) (HARRINGTON; PRESUHN; WIJNEN, 2002), é um exemplo de abordagem gerente-agente. A despeito da disseminação e compreensão do SNMP, sua utilização ainda está restrita a dispositivos de rede, sendo raramente utilizado para gerenciamento de sistemas e aplicações (SCHONWALDER; PRAS; MARTIN-FLATIN, 2003). Pode ser citado também o framework no qual o SNMP foi inspirado, o *Common Management Information Protocol* (CMIP) (Joint Technical Committee ISO/IEC, 1991), padrão de gerenciamento proposto pela *International Organization for Standardization* (ISO).

O modelo de gerenciamento centralizado, apesar de sua ampla disseminação, possui diversas limitações para os ambientes de rede atuais (PRAS et al., 2007). Uma das limitações desse modelo diz respeito à escalabilidade, pois o aumento do número de equipamentos gerenciados na rede acaba por elevar proporcionalmente a carga computacional e o tráfego de rede requerido na estação de gerenciamento, dificultando ou mesmo inviabilizando o gerenciamento adequado dos equipamentos. A centralização também apresenta problemas de Tolerância a Falhas, já que a estação de gerenciamento se torna um Ponto Único de Falhas (*Single Point of Failure* - SPoF) dentro do sistema de gerenciamento de redes. Outro ponto que pode ser ressaltado é a falta de flexibilidade, já que, no modelo centralizado, as funções de gerenciamento e as capacidades dos agentes são geralmente predefinidas e limitadas.

O modelo gerente-agente pode ser estendido para a definição de modelos mais sofisticados, possibilitando melhorias no desempenho desses itens. Dentre esses modelos, podem ser ressaltados aqueles nos quais é realizada a distribuição (em diferentes graus) das tarefas de gerenciamento, tais como o modelo de gerenciamento por delegação.

2.1.2 Gerenciamento Por Delegação

O modelo de Gerenciamento por Delegação (*Management by Delegation* - MbD) (GOLDSZMIDT; YEMINI, 1995) introduz funcionalidades de descentralização nas tarefas de gerenciamento. A descentralização é introduzida por meio do conceito de hierarquia de gerentes. A Figura 2.2 apresenta os elementos desse modelo.

Os elementos que constituem o modelo MbD são os gerentes de mais alto nível (*Top*

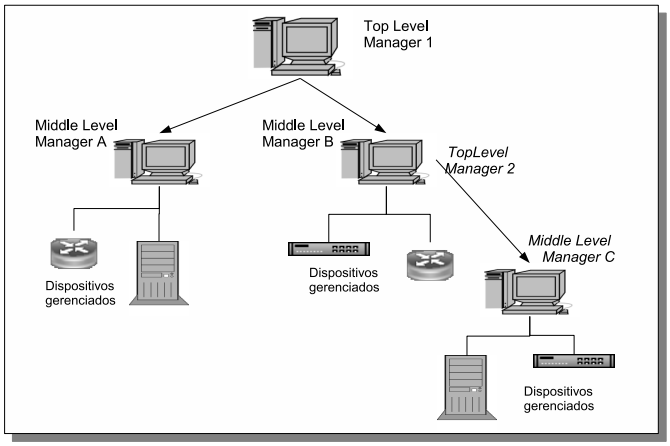

Figura 2.2: Modelo de gerenciamento por delegação

Level Manager - TLM), os gerentes de nível médio (*Middle Level Manager* - MLM) e os agentes. Conforme a Figura 2.2, um gerente pode assumir dois papéis ao mesmo tempo, ou seja, ser considerado um gerente de mais baixo nível (MLM B) em um contexto e ser um gerente de mais alto nível em outro contexto (TLM 2). Além disso, o modelo MbD especifica os mecanismos de delegação que possibilitam a representação desses diversos papéis aos gerentes.

Esse modelo introduz um bom nível de descentralização, uma vez que tarefas de gerenciamento são delegadas a diferentes gerentes que podem dividir, assim, a carga de processamento e de tráfego relativo às tarefas de gerenciamento. Entretanto, o modelo MbD não trata de diversas questões relevantes à descentralização das tarefas de gerenciamento, como, por exemplo, a utilização em múltiplos domínios administrativos.

2.1.3 Gerenciamento Baseado em Políticas

O objetivo do modelo de Gerenciamento de Redes baseado em Políticas (*Policy-Based Network Management* - PBNM) (SLOMAN, 1994) é controlar o comportamento do sistema de acordo com políticas previamente definidas. O modelo PBNM foi proposto tanto por indústrias como pelo meio acadêmico. Existem diversas arquiteturas e sistemas baseados nesse modelo, mas a definição mais aceita é a arquitetura PBNM proposta pelo IETF (*Internet Engineering Task Force*) (MOORE, 2003). O modelo PBNM proposto pelo IETF é composto por quatro elementos: Ferramenta de Políticas (*Policy Tool*), Repositório de Políticas (*Policy Repository*), *Policy Decision Point* (PDP) e *Policy Enforcement Point* (PEP). A Figura 2.3 mostra os componentes do modelo PBNM da IETF e seus relacionamentos.

O modelo PBNM também possui características distribuídas, já que o sistema pode ser formado por diferentes PDPs, elementos responsáveis pelas decisões relativas às tarefas de gerenciamento. A distribuição adequada não depende apenas do sistema gerenciado, mas também depende da arquitetura do sistema PBNM utilizado. Dessa forma, existe uma relação entre o ambiente de rede onde o sistema PBNM está localizado e sua própria organização.

Alguns aspectos, no entanto, dificultam uma efetiva distribuição dos sistemas PBNM, como, por exemplo, aspectos de consistência do estado das políticas (NOBRE; GRANVILLE, 2009a) (NOBRE; GRANVILLE, 2010). Em geral, os sistemas baseados em po-

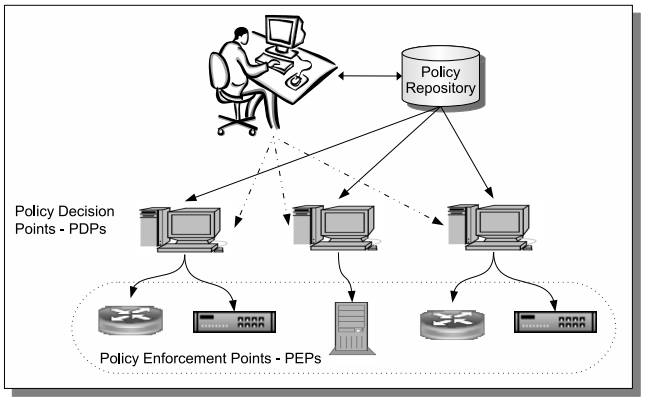

Figura 2.3: Modelo de gerenciamento de redes baseado em políticas

líticas tendem a apresentar certo grau de centralização, o que pode apresentar problemas relativos à escalabilidade e à robustez.

2.1.4 Gerenciamento de Redes baseado em P2P

A aplicação de tecnologias P2P (ANDROUTSELLIS-THEOTOKIS; SPINELLIS, 2004) tem sido proposta como uma ferramenta a ser utilizada em diferentes escopos, além do compartilhamento de arquivos, como foi inicialmente utilizada. Dentro deste contexto, o modelo de Gerenciamento de Redes de Computadores baseado em P2P (*P2P-based Network Management* - P2PBNM) (GRANVILLE et al., 2005) descreve um *overlay* de gerenciamento, o qual pode ser entendido como um conjunto de *peers* interligados no qual as tarefas de gerenciamento são executadas sobre o sistema físico gerenciado.

No modelo P2PBNM, *peers* têm um papel duplo: além de atuar como *peers* normais em um *overlay* P2P, executam tarefas de gerenciamento (BINZENHOFER et al., 2006). Dessa forma, o roteamento das mensagens necessárias às tarefas de gerenciamento é realizado na camada de aplicação. A realização do roteamento através da camada de aplicação possibilita o gerenciamento por meio de múltiplos domínios administrativos (FIORESE; SIMÕES; BOAVIDA, 2009).

Os sistemas P2P são reconhecidos pela sua escalabilidade (KARIMI et al., 2008), além de flexibilidade e dinamicidade. Além dessas características, a introdução de tecnologias P2P incorpora ao modelo P2PBNM outras características, tais como a replicação de serviços, distribuição de tarefas, auto-organização, tolerância a falhas e alta conectividade. A Figura 2.4 ilustra o modelo P2PBNM.

O modelo P2PBNM pode utilizar princípios do modelo MbD, como, por exemplo, a utilização de MLMs (GRANVILLE et al., 2005). No entanto, no modelo P2P os gerentes não estão estruturados em uma hierarquia rígida, podendo trocar informações entre entidades de mesmo nível hierárquico. Além disso, esse modelo também pode incorporar características do modelo PBNM. Dessa forma, é possível utilizar as características de conectividade inerentes a sistemas P2P para a distribuição de informações de gerenciamento (*e.g.*, políticas) por meio do sistema gerenciado.

Novas possibilidades de gerenciamento podem ser obtidas na utilização do modelo P2PBNM, tais como gerenciamento cooperativo baseado nas pessoas (MELCHIORS; GRANVILLE; TAROUCO, 2009) e balanceamento de carga para tarefas de gerenciamento (PANISSON et al., 2006). Além disso, alguns trabalhos indicam a utilização de tecnologia P2P para possibilitar a construção de sistemas de gerenciamento autonômico

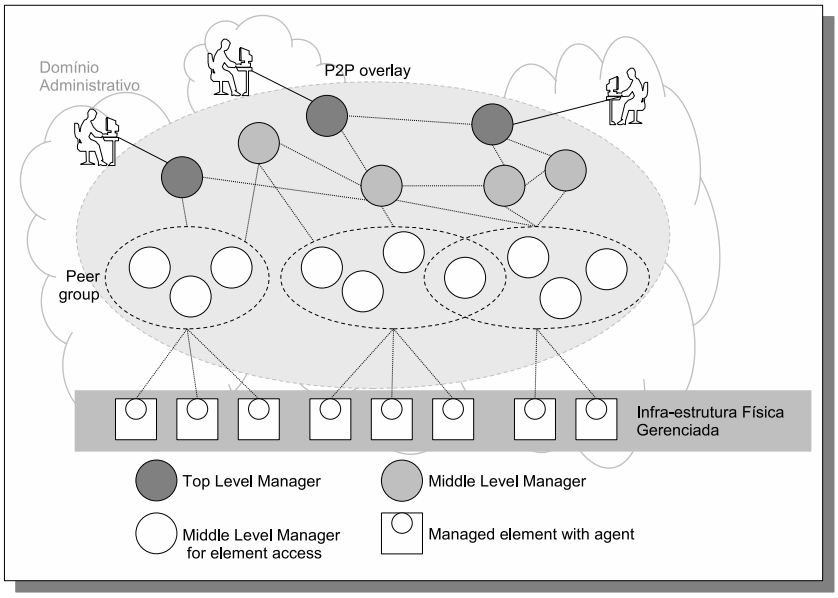

Figura 2.4: Modelo de Gerenciamento de Redes baseado em P2P

de redes descentralizados (MARQUEZAN et al., 2007) (KONSTANTINOU; YEMINI, 2009).

2.1.5 Gerenciamento Autonômico de Redes

A aplicação de conceitos de Computação Autonômica (*Autonomic Computing* - AC) em gerenciamento de redes, conhecida como Gerenciamento Autonômico de Redes (*Autonomic Network Management* - ANM), foi proposta como uma nova forma de controlar certos detalhes operacionais das tarefas dos administradores de rede, permitindo que as necessidades de intervenção humana permaneçam em níveis superiores. Sistemas Autonômicos se baseiam na metáfora biológica do sistema nervoso autônomo, tendo como principal conceito o autogerenciamento.

Um dos primeiros sistemas ANM propostos foi o *FOCALE* (*Foundation Observation Comparison Action Learn rEason*) (JENNINGS et al., 2007). Sua arquitetura foi constituída sobre um ambiente de gerenciamento de redes já estabelecido, e o principal ponto enfrentado nesse sistema é como lidar com um número significativo de diferentes dispositivos de rede e *softwares* de gerenciamento e possibilitar a troca de informação entre os mesmos. A solução adotada no FOCALE foi a utilização de modelos de ontologias, traduções baseadas em modelos e PBNM para apresentar características autonômicas como autoconhecimento.

Sistemas ANM como o *FOCALE* possuem características interessantes, no entanto, em geral, são necessários modelos de dados complexos para descrever suas informações, o que dificulta algumas operações. Também é interessante ressaltar que, usando abordagens centralizadas (ou mesmo fortemente acopladas) para ANM, algumas questões relativas ao gerenciamento de redes tradicional se mantêm, como escalabilidade e robustez. Isso faz alguns autores declararem que a decentralização é necessária para o oferecimento de capacidades completas de gerenciamento autonômico (JENNINGS et al., 2007).

2.1.6 Sistemas de Gerenciamento Autonômico de Redes baseados em P2P

Diferentes tecnologias podem ser empregadas como infraestrutura de sistemas ANM descentralizados. Uma possibilidade interessante é a utilização de *overlays* P2P, os quais incorporam características de sistemas de gerenciamento de redes baseado em P2P nos sistemas ANM (*e.g.*, suporte para colaboração em tarefas de gerenciamento). Este fato pode ser observado por um aumento no número de pesquisas relacionadas a essa utilização conjunta (EMANICS, 2010). Uma das maiores dificuldades é produzir modelos estáveis de cooperação para infraestruturas de gerenciamento altamente dinâmicas, mantendo as vantagens dos dois modelos (P2P e AC) (PRAS et al., 2007).

Diversos aspectos dos *overlays* P2P podem ser analisados para prover suporte eficiente para sistemas ANM. Uma analogia oriunda da Biologia que pode ser utilizada é o comportamento coerente e estável das células para a formação dos tecidos, os quais compõem os organismos multicelulares complexos. Tecidos possuem funcionalidades relacionadas à tolerância a falhas em sua operação e estratégias de comunicação (*i.e.*, sinalização entre células). De forma análoga, a operação de um *overlay* pode aumentar a disponibilidade de serviços replicando mensagens e operações entre diferentes *peers*.

Existem algumas pesquisas que investigam o ANM baseado em P2P. Nessas pesquisas, *peers* possuem algumas propriedades encontradas em Elementos de Gerenciamento Autonômico (*Autonomic Management Elements* - AME), e grupos de *peers* possuem algumas propriedades encontradas em Domínios de Gerenciamento Autonômico *Autonomic Management Domains* - AMD). AMEs e AMDs são conceitos utilizados para descrever os elementos constitutivos de sistemas de gerenciamento autonômico descentralizados (JENNINGS et al., 2007). Por questões de simplicidade, apenas algumas pesquisas serão apresentadas.

PBMAN (KAMIENSKI et al., 2006) une o PBNM tradicional com *overlays* P2P para gerenciar de forma autonômica *Ambient Networks* (AN). PBMAN é estruturado utilizando-se *super peers*, ou seja, em uma arquitetura hierárquica. Os *super peers* são responsáveis pela consistência do estado dos dados de gerenciamento, entre outras funções. Alguns dados de gerenciamento (*e.g.*, políticas) não são efetivamente distribuídos nesse sistema, e, por questões de confiabilidade, *super peers* são replicados.

A Plataforma Madeira (FALLON et al., 2007) é uma abordagem para ANM que utiliza o conceito de *Adaptive Management Components* (AMC), os quais são componentes executados nos dispositivos gerenciados. AMCs podem gerenciar elementos (nos quais estão sendo executados) e se comunicarem com outros AMCs (executados em outros dispositivo gerenciados) por meio de serviços de comunicação P2P. AMCs formam *cluster* de gerenciamento com *super peers*, atuando como *cluster heads*. Esses *super peers* são responsáveis pela consistência dos estados dos dados de gerenciamento, entre outras funções. A utilização de *cluster heads* produz dúvidas sobre a escalabilidade e robustez dos *clusters* de gerenciamento.

Self-Managed Cells (SMC) (LUPU et al., 2007) são propostas como uma arquitetura padronizada para aplicações de computação ubíqua, destinada a diferentes níveis de escala. Cada SMC é autônomo e utiliza políticas para direcionar suas decisões adaptativas. Entre as possíveis interações entre SMCs, os autores descrevem interações P2P. No entanto, de acordo com os níveis de abstração, não está claro se diferentes SMCs podem ser "*peers*". Um dispositivo gerenciado é logicamente conectado com apenas um SMC, dessa forma, dados de gerenciamento não são avaliados em uma abordagem P2P.

O ManP2P (PANISSON et al., 2006) é um exemplo de sistema de gerenciamento de redes baseado em P2P que está evoluindo para uma concepção autonômica pela imple-

mentação de módulos autonômicos que são executados nos *peers*. O ManP2P é inspirado no modelo MbD e baseado em uma abordagem orientada a serviços. Os módulos autonômicos são desenvolvidos para prover suporte a características *self-** e para se comunicarem com outros componentes do sistema ANM, quando necessário. Políticas são utilizadas para controlar parâmetros do *overlay*, e não existe distribuição de decisões de gerenciamento (MARQUEZAN et al., 2007). Não existe mecanismo interno para consistência de estado de dados de gerenciamento, assim, os autores propõem a utilização de repositórios externos.

Apesar das melhorias trazidas por sistemas ANM baseados em P2P (*e.g.*, maior disponibilidade do sistema de gerenciamento), ainda existem alguns problemas a serem resolvidos. O estado dos dados de gerenciamento em diferentes *peers* de um sistema ANM baseado em P2P pode se tornar inconsistente na operação do *overlay* devido a falhas ou falta de sincronização. Além disso, a manutenção da consistência do estado dos dados de gerenciamento deve ser realizada mantendo as características de escalabilidade e robustez dos *overlays* P2P. Essa manutenção de consistência é igualmente importante para qualquer sistema ANM descentralizado apresentar um comportamento coerente global. A consistência do estado dos dados de gerenciamento entre os *peers* é normalmente realizada com alguma centralização, usando *super peers* (KAMIENSKI et al., 2006) (FALLON et al., 2007) ou repositórios externos (MARQUEZAN et al., 2007), o que desperdiça oportunidades de *overlays* P2P (e.g, robustez).

O cenário enfrentado pelo sistemas ANM baseado em P2P para manter a consistência dos estados dos dados de gerenciamento é similar ao cenários enfrentados em diferentes Sistemas Multiagente (*Multi-Agent Systems* - MAS). Agentes precisam avaliar e manter a integridade das informações trocadas entre os mesmos e conflitos sobre conhecimento contraditório podem acontecer durante a comunicação. Além disso, agentes devem executar suas tarefas de forma assíncrona. Assim, técnicas oriundas de MAS podem ser utilizadas para aprimorar a consistência dos estados das informações de gerenciamento em sistemas ANM baseados em P2P.

2.2 Trabalhos Relacionados

2.2.1 Abordagem Multiagente para a Computação Autonômica

Alguns autores argumentam que a abordagem dos Sistemas Multiagente (*Multi-Agent Systems* - MAS) é apropriada para Computação Autonômica (*Autonomic Computing* - AC) (TESAURO et al., 2004) (HU et al., 2004). Muitas técnicas desenvolvidas na comunidade dos MAS, tais como aquelas relacionadas com formação automática de grupos, comportamento emergente, adaptação multiagente e coordenação de agentes, poderiam ser adaptadas para solucionar os desafios práticos enfrentados pela AC (TESAURO et al., 2004).

Técnicas oriundas dos MAS podem proporcionar a base para a construção de sistemas autonômicos distribuídos e cooperativos, tais como sistemas de Gerenciamento Autonômico de Redes (*Autonomic Network Management* - ANM) descentralizados. Em termos gerais, tecnologias relacionadas a agentes têm sido identificadas como um componente bastante significativo para a implementação de autonomia em diversos sistemas (HU et al., 2004). Além disso, MAS podem se beneficiar de serviços de comunicações de baixo nível, tais como os oferecidos por arquiteturas P2P.

2.2.2 Estudo das combinações entre Sistemas Multiagente e *Overlays* P2P

Muitos autores já apontaram as vantagens na aproximação dos paradigmas P2P e MAS, considerando que esses conceitos são complementares e que essa integração pode auxiliar na resolução de carências de ambas as tecnologias (MORO; OUKSEL; SARTORI, 2003). Ambos são definidos como auto-organizáveis, operando em ambientes potencialmente dinâmicos, com frequentes entradas e saídas de *peers*/agentes.

Conceitos de MAS podem permitir que os sistemas P2P utilizem descrições de sistemas complexos com um maior nível de abstração, mediante metáforas que capturem melhor situações reais. Por exemplo, é possível controlar aspectos de cooperação e comunicação entre *peers* por meio de agentes que neles estejam. Agentes podem, por exemplo, priorizar tarefas, procurar por informações locais, determinar comportamentos anômalos e produzir respostas adequadas e negociar e colaborar para objetivos específicos (MORO; OUKSEL; SARTORI, 2003). Além disso, MAS podem oferecer conceitos e técnicas proveitosas para infraestruturas P2P na modelagem de aplicações e no nível de definições (*e.g.*, ontologias para descrever recursos de rede em uma forma semanticamente significativa).

MAS utilizam, em geral, sistemas distribuídos tradicionais como infraestrutura e a migração para uma infraestrutura P2P é uma boa alternativa para compensar as dificuldades enfrentadas nos sistemas distribuídos de grande escala, ubíquos ou móveis (*e.g.*, dinamismo). A aplicação de tecnologia P2P traz maior rendimento, escalabilidade, disponibilidade e confiabilidade em comparação com os sistemas distribuídos tradicionais. Além disso, a aplicação de tecnologia P2P pode oferecer domínios de aplicação populares prontos, técnicas sofisticadas de implementação e componentes de infraestrutura para a construção de MAS (KOUBARAKIS, 2003).

Existem diversas formas de realizar a integração de MAS e P2P. Dentre essas formas, pode-se realizar a operação conjunta por meio da utilização de um *middleware* entre camadas relativas à infraestrutura P2P e o MAS. Dessa forma, essas camadas podem ser designadas da seguinte forma (OOI et al., 2003): *camada agente* para realizar os serviços relacionados a aplicações no *peer*; *plataforma P2P* para controlar a comunicação e as estratégias de roteamento de mensagens; *camada intermediária* para controlar a comunicação entre as camadas P2P e agente.

A integração entre MAS e P2P pode trazer, no entanto, algumas dificuldades (MORO; OUKSEL; SARTORI, 2003) (KOUBARAKIS, 2003). Entre essas dificuldades, podem ser ressaltadas a possibilidade do sistema, como um todo, tornar-se mais "pesado"e uma grande disparidade entre as linguagens e protocolos utilizados entre MAS (*e.g.*, *Knowledge Query Manipulation Language* - KQML e *Agent Communication Language* - ACL) e P2P.

2.2.3 Sistemas de manutenção da Verdade

Sistemas de manutenção da verdade (*Truth-Maintenance Systems* - TMSs) foram propostos para manter a integridade de Bases de Conhecimento (*Knowledge Bases* - KBs). Um TMS controla a estrutura lógica do conjunto de crenças de agentes. Uma crença participa do conjunto atual de crenças se a mesma possui razões válidas (DOYLE, 1979). A responsabilidade de avaliar a consistência lógica das proposições é de um resolvedor de problemas (Problem Solver - PS), sendo que o mesmo envia as suas conclusões e as respectivas justificativas ao módulo TMS. Assim, o TMS registra e associa a cada conclusão os seus fundamentos, garantindo a consistência. A figura 2.5 demonstra a relação entre o

TMS e o PS.

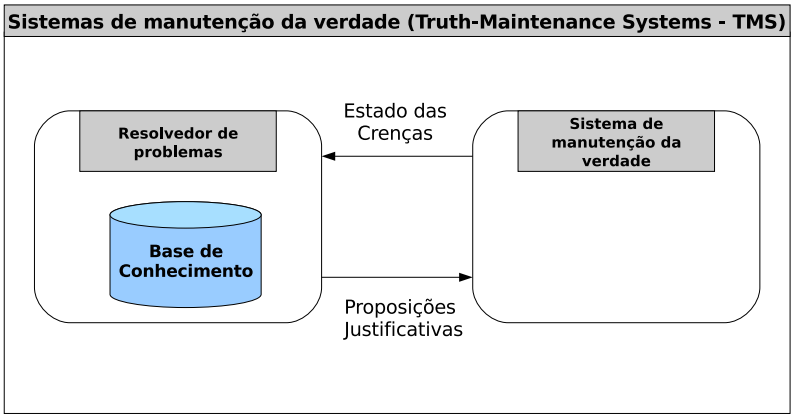

Figura 2.5: Sistemas de manutenção da verdade

TMS proporcionam poder considerável utilizando poucos recursos computacionais (KAGAL; HANSON; WEITZNER, 2008). As origens desses sistemas foram propostas ainda nos anos 70 para a resolução de situações em sistemas monoagente (DOYLE, 1979). Dessa forma, apesar de não serem conhecidos fora da comunidade de Inteligência Artificial, TMS são utilizados em diferentes contextos, tais como sistemas de recomendação (LORENZI, 2007), esquemas baseados em ontologias (CLARK; MCCABE, 2007), sistemas baseados em políticas (KAGAL; HANSON; WEITZNER, 2008) e sistemas de gerenciamento de redes (NOBRE; GRANVILLE, 2009a) (NOBRE; GRANVILLE, 2009b) (NOBRE; GRANVILLE, 2010).

TMS foram estendidos para versões multiagente, dentre as quais pode ser ressaltada a versão conhecida como *Distributed Truth-Maintenance Systems* (DTMS) (HUHNS; BRIDGELAND, 1991). Em um MAS, os agentes necessitam de meios para manter a integridade de suas KBs, face a trocas de mensagens com outros agentes. Essa manutenção de integridade pode ser realizada por um TMS multiagente. De forma análoga, durante a operação de sistemas ANM baseados em P2P, *peers* devem possuir meios de manter a integridade dos estados de seus dados de gerenciamento, na presença de troca de mensagens dentro do *overlay* de gerenciamento. Esta similaridade indica o uso de TMS multiagente em ANM baseado em P2P como uma possibilidade interessante (NOBRE; GRANVILLE, 2009a).

2.2.4 Consistência de informações compartilhadas em sistemas distribuídos

Serviços que oferecem suporte à consistência de informações compartilhadas podem ser utilizados como um bloco básico para a construção de aplicações distribuídas. Sistemas ANM baseados em P2P, como aplicações distribuídas, podem se apropriar de características encontradas nesses serviços. A consistência dos dados de gerenciamento compartilhados possui um papel bastante importante na operação coerente de sistemas ANM descentralizados, tais como aqueles baseados em P2P.

Em termos genéricos, qualquer forma de aplicação distribuída fracamente acoplada pode se beneficiar de mecanismos para consistência de informações compartilhadas. Assim, são apresentados alguns serviços, que apesar de não serem desenvolvidos especificamente para sistemas de gerenciamento, possuem algumas características análogas ao serviço proposto na presente dissertação.

O *ZooKeeper* (HUNT, 2010), parte integrante do projeto *Hadoop*, é um serviço de coordenação para aplicações distribuídas, o qual pode ser utilizado para a implementação de serviços de alto nível para sincronização, difusão de dados e esquemas *publish-subscribe*. O serviço é oferecido por meio de um conjunto simples de primitivas e utiliza um modelo de dados semelhante às árvores de diretórios dos sistemas de arquivos, replicadas em diversos servidores. O *ZooKeeper* utiliza um esquema tradicional cliente-servidor, cada cliente se conecta apenas a um servidor. Dessa forma, operações de leitura são respondidas por meio de consultas à base de informações de cada servidor. No entanto, em operações de escrita é utilizado um servidor "líder" (*i.e.*, um banco de dados centralizado), para manutenção de consistência. Dessa forma, esse serviço possui as vulnerabilidades associadas a sistemas centralizados.

O *Astrolabe* (VAN RENESSE; BIRMAN; VOGELS, 2003) é um serviço distribuído de gerenciamento de informações, o qual monitora dinamicamente o estado de um grupo de recursos distribuídos, relatando agregações dessas informações para os usuários. Esse serviço implementa uma estrutura de dados replicada acessada por meio de consultas SQL. O *Astrolabe* é implementado por um *overlay* P2P, no qual todos os *peers* executam um agente *Astrolabe* (*i.e.*, em uma abordagem MAS). *Astrolabe* foi desenvolvido utilizando modelos de dados simples. Além disso, sua operação é focada em aplicações nas quais há predomínio de operações de leitura. Um das aplicações proposta pelos autores para esse serviço é o monitoramento de rede, uma tarefa tradicional do gerenciamento de redes.

Scalable Distributed Information Management System (SDIMS) (YALAGANDULA; DAHLIN, 2004) é um serviço para agregar informações sobre sistemas distribuídos de larga escala. O serviço provê flexibilidade por meio de uma API simples que permite que as aplicações controlem a propagação de leituras e escritas. O serviço foi construído utilizando conceitos oriundos do *Astrolabe* (VAN RENESSE; BIRMAN; VOGELS, 2003) em conjunto com *Distributed Hash Tables* (DHT). No entanto, assim como na maioria das aplicações que utilizam DHT, questões relativas à consistência e replicação são um desafio conhecido.

Os serviços apresentados demonstram características interessantes para a consistência de informações compartilhadas em sistemas distribuídos. No entanto, esses serviços possuem vulnerabilidades que os tornam pouco apropriados para sistemas ANM baseados em P2P, tais como centralização (HUNT, 2010), modelos simples de dados (VAN RENESSE; BIRMAN; VOGELS, 2003), e questões relativas a replicação (YALAGANDULA; DAHLIN, 2004).

A consistência dos estados das dados de gerenciamento ainda é a um tópico em aberto em sistemas ANM baseados em P2P. É necessário realizar a manutenção da consistência mantendo as características desejáveis de escalabilidade e robustez dos *overlays* P2P. Além disso, neste contexto, é interessantes que essa manutenção apresente características de serviços para consistência de informações compartilhadas. A manutenção da integridade das KBs entre agentes trazida por TMS multiagente parece ser uma direção válida para introduzir a manutenção de consistência (NOBRE; GRANVILLE, 2009a).

3 PROPOSTA

Sistemas de Gerenciamento Autonômico de Redes (*Autonomic Network Management* - ANM) baseados em *Peer-to-Peer* (P2P) necessitam compartilhar dados de gerenciamento entre os diversos *peers* que os compõem. No presente trabalho, *dado de gerenciamento* é definido com uma informação de gerenciamento descrita em uma forma definida (*i.e.*, usando uma linguagem específica). Nesses sistemas, espera-se que as fontes para os dados de gerenciamento apresentem desafios para sua utilização (*e.g.*, ambientes de rede altamente dinâmicos). Apesar desses desafios, é necessário que se evite inconsistências potenciais no estado dos dados de gerenciamento entre os *peers*.

Sistemas ANM utilizam diversos grupos de dados de gerenciamento em sua operação, os quais podem ser integrados em *Bases de Conhecimento* (*Knowledge Bases* - KB). A manutenção de consistência das KBs de gerenciamento é um dos principais desafios relacionados aos sistemas ANM descentralizados, tais como sistemas ANM baseados em P2P. Nesses sistemas, essas bases de conhecimento são distribuídas (e compartilhadas) entre os elementos que formam esses sistemas, conhecidos como Elementos de Gerenciamento Autonômico (*Autonomic Management Elements* - AME). Em um sistema ANM baseado em P2P, *peers* possuem características análogas a AMEs. Além disso, AMEs que compartilham dados específicos são integrados em um Domínio de Gerenciamento Autonômico (*Autonomic Management Domain* - AMD). Em um sistema ANM baseado em P2P, grupos de *peers* possuem características análogas a AMDs.

Dados de gerenciamento manipulados por sistemas ANM devem permitir sua utilização em procedimentos de automatização e otimização. Um ponto a ser ressaltado é que a execução de operações complexas em bases distribuídas de conhecimento de gerenciamento amplia o surgimento de desafios relativos à manutenção de consistência. Técnicas de raciocínios e aprendizagem em ontologias distribuídas para gerenciamento de redes podem ser citadas como exemplos dessas operações. Dessa forma, os procedimentos realizados em um sistema ANM descentralizado (e.g, sistema ANM baseado em P2P) aumentam os desafios relacionados à consistência do estado dos dados de gerenciamento.

A presente proposta visa atingir os requisitos de consistência de estado dos dados de gerenciamento de sistemas ANM baseados em P2P. O mecanismo proposto introduz funcionalidades de *Sistema de Manutenção da Verdade* (*Truth Maintenance Systems* - TMS) em uma versão multiagente (HUHNS; BRIDGELAND, 1991), por meio da utilização de um *módulo de manutenção de consistência*, que é executado em cada um dos *peers* de um sistema ANM baseado em P2P. Dessa forma, diferentes *peers* possuiriam algumas características análogas a agentes e o sistema ANM baseado em P2P agregaria algumas características de Sistemas Multiagente (*Multi-Agent Systems* - MAS). Conforme análises da literatura, os únicos estudos que incorporam funcionalidades de TMS no gerenciamento de redes foram conduzidos por *Nobre e Granville* (NOBRE; GRANVILLE, 2009a)

(NOBRE; GRANVILLE, 2009b) (NOBRE; GRANVILLE, 2010).

TMSs multiagente são extensões multiagente para TMSs. O objetivo da utilização de TMSs é manter a integridade de KBs (DOYLE, 1979). Em um TMS multiagente, existem múltiplos agentes e cada um deles possui seu próprio TMS. TMSs mantêm a integridade das KBs executando *revisão e troca de crenças* sobre os dados dessas KBs. Uma crença faz parte do conjunto atual de crenças se a mesma tem razões válidas (*i.e.*, a crença é bem fundamentada).

A avaliação da consistência lógica de uma KB é realizada pelo Resolvedor de Problemas (*Problem Solver* - PS) (HUHNS; BRIDGELAND, 1991). O PS envia suas crenças e respectivos fundamentos ao TMS. O TMS, então, registra e associa a cada crença os fundamentos enviados, garantindo a consistência dos estados das mesmas. Por exemplo, em um sistemas baseado em políticas, as funções relativas ao PS são executadas pelo componente de software que realiza o processamento de políticas.

A troca de crenças sobre os dados de gerenciamento é realizada de forma assíncrona e considera-se que a troca de mensagens não é confiável. São bem conhecidos os desafios impostos à manutenção da consistência de dados compartilhados em sistemas distribuídos assíncronos não confiáveis (FISCHER; LYNCH; PATERSON, 1985). Dessa forma, o modelo de consistência utilizado é não determinístico, em outras palavras, é utilizada um noção “fraca” de consistência. Esse modelo foi adotado em função de questões relativas à escalabilidade, robustez e disseminação de atualizações. Dada uma crença “X” que depende de uma outra crença “Y”, quando uma atualização é realizada em “Y”, essa atualização será refletida posteriormente em “X”. Alguns autores chamam uma noção similar de “*eventual consistency*” (VAN RENESSE; BIRMAN; VOGELS, 2003).

A computação de crenças ocorre de forma concorrente dentro do grupo de *peers*, assim, é possível que o estado de um dado (inferido por diferentes *peers*) seja baseado em um conjunto diferente de crenças. Diferentes *peers* participantes de um grupo de *peers* não possuem garantias de terem cópias idênticas do conjunto atual de crenças, mesmo se esses *peers* forem consultados no mesmo momento, assim como nem todos os *peers* têm garantias de perceber cada uma das atualizações do conjunto atual de crenças. Para alcançar consistência, todas as mudanças de crenças devem ser replicadas entre todos os *peers* de um grupo de *peers*. Mesmo assim, ainda pode ser necessário um tempo significativo para que todos os *peers* do grupo de *peers* se tornem consistentes.

Algumas hipóteses são assumidas na presente proposta. As tarefas de gerenciamento podem ser executadas completamente em cada um dos *peers*, dessa forma, a proposta deve fornecer habilidades de coordenação para sistemas ANM baseados em P2P (aspectos de cooperação não serão discutidos). É proposta, ainda, a hipótese de que não existem recursos compartilhados globais, ao menos, em tempo de operação. Além de ser o meio de troca de mensagens, o *overlay* P2P deve prover suporte à organização de grupos, por meio dos serviços de gerenciamento disponibilizados. Assim, *peers* que oferecem um serviço específico são organizados em um grupo (sem intervenção humana) e podem participar de diversos grupos de acordo com seus serviços oferecidos (MARQUEZAN et al., 2007).

3.1 Justificativas para dados de gerenciamento

Uma das possíveis alternativas para implementar funcionalidades TMS multiagente é a utilização de *justificativas* (HUHNS; BRIDGELAND, 1991). Em um *TMS baseado em justificativas*, um dado é acreditado se o mesmo possui justificativas válidas (*i.e.*, razões válidas), ou seja, a crença sobre o estado de um dado é fundamentada nas suas justificati-

vas. As justificativas são produzidas de acordo com as informações disponíveis, em uma forma definida (que possa ser entendida pelo TMS) (DOYLE, 1979). No presente trabalho, propõe-se que as crenças sejam compartilhadas entre diferentes *peers* que formam o sistema ANM baseado em P2P.

Em sua especificação original (DOYLE, 1979), os TMS não utilizam uma lista definida de possíveis justificativas para um dado. No entanto, a fim de simplificar questões de implementação e análise, a presente proposta define que o módulo de manutenção de consistência deve receber o dado e sua lista de justificativas necessárias (estrutura) como mostrado na Figura 3.1. O dado e sua lista de possíveis justificativas deve ser suprido por um administrador de redes ou um sistema especialista. A presença de justificativas é baseada em informações de gerenciamento processadas pelo sistema ANM baseado em P2P.

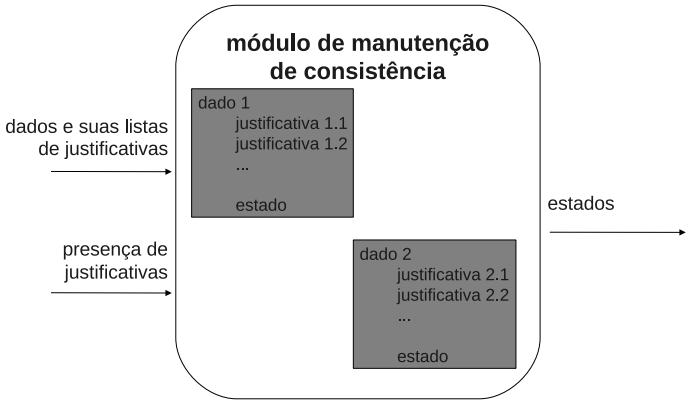

Figura 3.1: Módulo de Manutenção de Consistência

A utilização de justificativas pode auxiliar no alinhamento dos *peers* em relação a objetivos gerais do sistema ANM baseado em P2P, já que cada um dos *peer* mantém sua estrutura de dados localmente (*i.e.*, dados e suas respectivas justiticativas). No entanto, como os *peers* podem possuir conjuntos distintos de dados, os mesmos não necessitam ser homogêneos. Dessa forma, os dados de um *peer* podem ser divididos em duas classes dentro de um grupo de *peers*: dados compartilhados, sobre os quais os *peers* desse grupo trocam crenças; e dados privados, que são apenas armazenados nas KBs dos *peers*. A Figura 3.2 (figura adaptada (HUHNS; BRIDGELAND, 1991)) demonstra a relação entre dados compartilhados e privados dentro de um grupo de *peers*.

A Figura 3.2 mostra um grupo de *peers* que é composto por 3 *peers* em um momento específico. Além disso, são demonstrados os dados que cada um desses *peers* possui. Assim, apenas o dado "Q" é compartilhado dentro do grupo de *peers*, sendo os outros dados representados (*e.g.*, "R" e "T") privados em relação a *peers* determinados. Somente os dados compartilhados precisam ter homogeneidade em seus estados e, consequentemente, utilizar manutenção da consistência dos estados. Assim, de acordo com o grupo de *peers* apresentado na Figura 3.2, apenas crenças sobre o dado "Q" seriam trocadas nesse grupo.

Justificativas também podem ser utilizadas para prover explicações sobre dados espe-

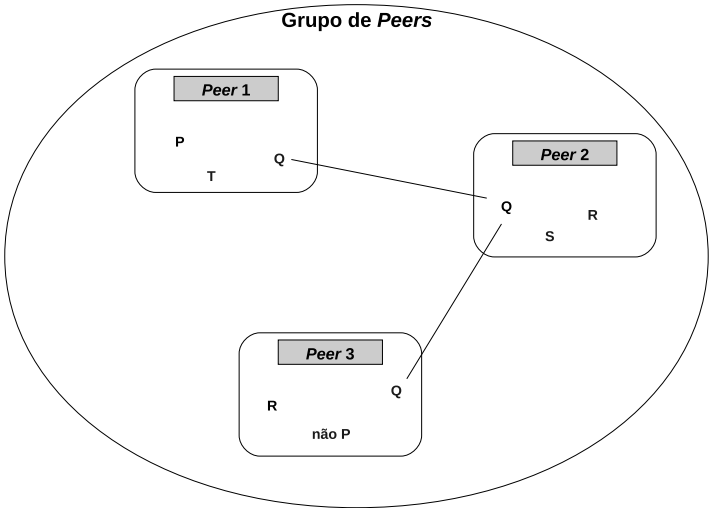

Figura 3.2: Dados compartilhados e privados

cíficos aos usuários (KAGAL; HANSON; WEITZNER, 2008). A utilização de justificativas para dados de gerenciamento demonstra esse poder explanatório (NOBRE; GRANVILLE, 2009b), possibilitando que administradores de rede utilizem essa informação a fim de aprimorar a execução de tarefas de gerenciamento e a compreensão sobre o significado desses dados. Assim, essa utilização pode aumentar a confiança nos resultados das tarefas de gerenciamento (*e.g.*, aplicação de políticas) (KAGAL; HANSON; WEITZNER, 2008).

A utilização de justificativas para dados de gerenciamento pode ser demonstrada em um exemplo, em que a ativação (crença) de uma política (dado) de Qualidade de Serviço (*Quality of Service* - QoS) pode ser justificada por um comando de um administrador de rede (justificativa) e um sinal assíncrono oriundo de um dispositivo gerenciado (justificativa). O código abaixo apresenta uma possível representação interna do dado e de suas justificativas usando o padrão *ISO Prolog* (SCOWEN, 1995). Nesse exemplo, justificativas e suas presenças são representadas por fatos (notação Prolog), e o dado é representado por uma regra (notação Prolog). O estado do dado "qos_pol" ("*Quality of Service Policy*") é definido pela regra "datum(qos_pol)", a qual define que se as justificativas "adm_cmd" ("*Administrator Command*") e "async_sig" ("*Asynchronous Signal*") estão presentes, esse dado é acreditado.

```
justification(adm_cmd).
justification(async_sig).
justificationIsPresent(adm_cmd).
justificationIsPresent(async_sig).
datum(qos_pol) :-
 justificationIsPresent(adm_cmd),
 justificationIsPresent(async_sig).
```

Os estados associados a um dado são, "in" (acreditado) ou "out" (desacreditado), de acordo com suas justificativas (DOYLE, 1979). Um dado é rotulado como "out" quando não está presente, ao menos, uma de suas justificativas associadas. Neste trabalho assume-se que não existem contradições nas justificativas fornecidas, então, um dado é desacreditado apenas se algumas de suas justificativas não estiver presente.

Justificativas podem ser geradas por processos do próprio *peer* ou recebidas por meio dos serviços de comunicação do *overlay* P2P. Dessa forma, o estado “in” pode assumir duas opções: “internal”, quando o dado é acreditado e tem apenas justificativas válidas internas; e “external”, quando o dado é acreditado e possui, pelo menos, uma justificativa válida externa (HUHNS; BRIDGELAND, 1991). Na presente proposta, “justificativa externa” significa que a mesma foi recebida por meio dos serviços de comunicação do *overlay* P2P.

O exemplo anterior, a ativação de uma política de QoS, pode ser atualizado para demonstrar a representação de estados adicionais para um dado. A fim de habilitar esses estados, o fato “justificationIsPresent” é substituído pela regra “justificationIsPresent” e também é criada uma nova regra, “datumIsInternal”. Assim, a presença de uma justificativa é inferida por fatos “generated” (justificativa gerada internamente) ou “received” (justificativa recebida por meio dos serviços de comunicação do *overlay* P2P). O código abaixo demonstra uma nova representação do dado, a qual inclui um nova versão de “justificationIsPresent” e a implementação de “datumIsInternal(qos_pol)” (para determinar se o dado está “internal” ou “external”).

```
justification(adm_cmd).
justification(async_sig).
justificationIsPresent(X) :- generated(X).
justificationIsPresent(X) :- received(X).
datum(qos_pol) :-
 justificationIsPresent(adm_cmd),
 justificationIsPresent(async_sig).
datumIsInternal(qos_pol) :-
 generated(adm_cmd),
 generated(async_sig).
```

O estado atual de um dado pode ser consultado durante a operação do sistema ANM baseado em P2P. Esse estado é descrito mediante a presença das justificativas e da fonte dessas presenças. O código abaixo demonstra uma resposta produzida pelo módulo de manutenção de consistência para o exemplo anterior, a ativação de uma política de QoS.

```
qos_pol:internal (adm_cmd:mod async_sig:mod)
```

A resposta produzida pelo módulo de manutenção de consistência indica que o estado do dado de gerenciamento “qos_pol” é “in” (*i.e.*, acreditado) e marcado com a opção “internal”. A ausência de alguma das justificativas faria o estado do dado ser apresentado como “:ou” (*i.e.*, desacreditado). Além disso, a resposta mostra que a presença das justificativas foi gerada internamente (indicada pelo “:mod” no trecho de código). Se a presença de alguma das justificativas fosse verificada pelo recebimento de mensagens de mudanças de crenças, essa presença seria indicada por “:msg”, ao lado dessa justificativa, e o estado do dado seria marcado como “external”.

Justificativas também podem ser utilizadas de forma hierárquica, assim um dado pode ser utilizado como justificativa para outros dados. Neste caso, a presença deste tipo de justificativa é controlado pelo estado do dado, em outras palavras, se o dado está “in”, o mesmo aparece como uma justificativa presente; e se o dado está “out”, o mesmo aparece como uma justificativa ausente. Essa utilização permite a representação de dados mais complexos.

O exemplo anterior, a ativação de uma política de QoS, pode ser alterado para demonstrar a utilização de justificativas hierárquicas. A justificativa “adm_cmd” (“*Administrator Command*”) pode ser modificada a fim de englobar dois diferentes comandos de um administrador humano. Assim, será criado o dado “adm_cmd” e esses diferentes comandos

serão representados pelas novas justificativas “adm_cmd1” e “adm_cmd2”, conforme o seguinte trecho de código.

```
justification(adm_cmd1).
justification(adm_cmd2).
justificationIsPresent(X) :- generated(X).
justificationIsPresent(X) :- received(X).
datum(adm_cmd) :-
 justificationIsPresent(adm_cmd1),
 justificationIsPresent(adm_cmd2).
datumIsInternal(adm_cmd) :-
 generated(adm_cmd1),
 generated(adm_cmd2).
```

O novo dado “adm_cmd” será, então, utilizado como justificativa para o dado “qos_pol”, por meio de uma modificação na regra “datum(qos_pol)”. Além disso, a regra “datumIsInternal(qos_pol)” é modificada para que a opção “internal” seja marcada corretamente. Assim, se as justificativas de um dado são geradas internamente, então esse dado é considerado como uma justificativa também gerada internamente. O trecho de código abaixo demonstra essas modificações.

```
datum(qos_pol) :-
 datum(adm_cmd),
 justificationIsPresent(async_sig).
datumIsInternal(qos_pol) :-
 datumIsInternal(adm_cmd),
 generated(async_sig).
```

A utilização de justificativas descrita nesta dissertação não contempla a funcionalidade de representação de justificativas que identificam ausências em relação a crenças. O trabalho inicial com TMS define duas listas: lista “IN”, em que estão representadas as crenças acreditadas; e lista “OUT”, onde estão representadas as crenças desacreditadas (DOYLE, 1979). Dessa forma, a representação de uma ausência é definida como parte de uma lista “OUT”. Na presente proposta, há apenas uma lista de justificativas, sendo a mesma relacionada com as crenças acreditadas (lista “IN”). No entanto, a definição de uma justificativa que identifica uma ausência pode ser realizada pelo administrador humano, utilizando-se, indiretamente, do significado das justificativas. Assim, por exemplo, a ausência de um comando de um administrador poderia ser descrita por um fato “adm_cmd_off” (“*Administrator Command Off*”).

3.2 Arquitetura dos Peers

A arquitetura dos *peers* que compõem o sistema ANM baseado em P2P pode ser vista como um *container* que comporta um ou mais *módulos de serviços de gerenciamento*. *Módulos de serviços de gerenciamento* executam tarefas de gerenciamento (*e.g.*, coleta de estatísticas) em cada *peer*, e, nessas tarefas, os módulos produzem dados de gerenciamento, criando *bases de conhecimento de gerenciamento*.

Cada *peer* dentro de um grupo de *peers* possui o mesmo nível hierárquico em relação aos demais, dessa forma, não há necessidade de esquemas de eleição ou representações especiais (VAN RENESSE; BIRMAN; VOGELS, 2003). Além disso, *peers* podem aparecer e desaparer durante a operação normal do *overlay*, comportamento que não é esperado dos elementos constitutivos dos sistemas de gerenciamento tradicionais.

A introdução de justificativas para os dados de gerenciamento foi implementada na presente dissertação por meio de um módulo adicional. Esse módulo, o *módulo de manutenção da consistência*, tem como função gerenciar o conjunto de crenças sobre dados de

gerenciamento em cada *peer*. Esse módulo trabalha associando dados de gerenciamento e suas respectivas justificativas. Quando ocorre uma mudança de crença (*i.e.*, mudança na presença de uma justificativa), o módulo de manutenção da consistência utiliza os serviços de comunicação do *overlay* P2P para difundir a mudança. A Figura 3.3 mostra a relação entre módulo de manutenção da consistência, módulos de serviços de gerenciamento e serviços de comunicação do *overlay* P2P.

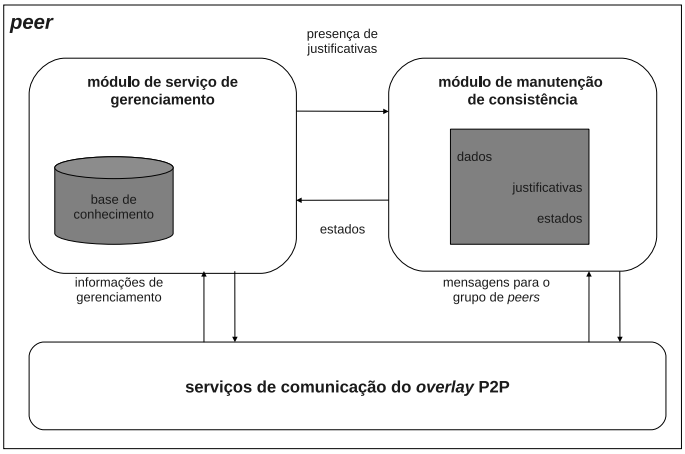

Figura 3.3: Arquitetura dos *peers*

Quando a presença de uma justificativa é modificada (internamente ou por recepção de mensagens), o módulo de manutenção da consistência executa os seguintes passos: retira os rótulos do estado dos dados de gerenciamento, atualiza a informação sobre a presença de justificativas e coloca novamente os rótulos de acordo com as novas restrições. Os serviços de comunicação do *overlay* P2P são potencialmente utilizados para difundir as mudanças, as quais podem alterar crenças de outros *peers*. Como descrito anteriormente, não existem recursos centralizados de armazenamento, assim, os dados e suas justificativas são replicados em cada um dos *peers* do grupo de *peers*.

Dados de gerenciamento são produzidos em diferentes linguagens. *A priori*, o módulo de manutenção da consistência não define uma linguagem particular para a definição das informações de gerenciamento. No entanto, a representação interna dos dados de gerenciamento e suas justificativas deve ser única entre os *peers* de um grupo de *peers*. A presente proposta apenas introduz uma possível representação, utilizando fatos e regras, conceitos da linguagem ISO PROLOG.

Os dados de gerenciamento podem ser divididos em duas classes: compartilhados (entre os *peers* de um grupo de *peers*) e privados (dados não compartilhados). *Peers* não precisam ser homogêneos, assim, alguns dados podem ser privados e não necessitarem ser consistentes com outros *peers*. Por outro lado, dados de gerenciamento compartilhados nos *peers* necessitam de mecanismos de manutenção de consistência. Os módulos de serviços de gerenciamento devem informar suas crenças internas sobre os dados de gerenciamento ao módulo de manutenção da consistência para utilizar esse mecanismo.

A responsabilidade da consistência lógica de um dado e suas justificativas não está incluída entre as operações executadas pelo módulo de manutenção de consistência. Esse módulo verifica se as justificativas definidas para um certo dado estão presentes e, usando esta informação, controla o estado do dado. Uma vantagem desta abordagem é que a ope-

Método	Descrição	Argumentos
justification_presence	presença de justificativa	nome da justificativa
justification_absence	ausência de justificativa	nome da justificativa
justification_query	consulta de presença de justificativa	nome da justificativa
datum_query	consulta do estado do dado	nome do dado

Tabela 3.1: Métodos relativos a mudanças de crenças na utilização do módulo de manutenção da consistência

ração do módulo de manutenção da consistência é apenas desencadeada quando ocorre uma mudança de crença sobre dados compartilhados, diminuindo a necessidade de recursos computacionais, os quais podem ser utilizados para a execução das tarefas de gerenciamento no *peer*.

Os módulos de serviços de gerenciamento são também responsáveis por consultar e requisitar serviços ao módulo de manutenção da consistência, de acordo com as demandas dos seus processos. Alguns serviços similares empregam estratégias diferentes, tais como esquemas "*publish/subscribe*" (VAN RENESSE; BIRMAN; VOGELS, 2003) ou "*watches*" (mudanças desencadeiam a transmissão de um pacote) (HUNT, 2010). Essas estratégias poderiam ser implementadas no módulo de manutenção da consistência, no entanto, inicialmente, a consulta e requisição de serviços está a cargo dos módulos de serviços de gerenciamento.

É importante ressaltar que existe apenas um módulo de manutenção da consistência dentro de um *peer*, assim, o mesmo não é restrito a módulos de serviços de gerenciamento específicos. Logo, todo módulo de serviços de gerenciamento (que utilizar serviços de manutenção da consistência) em um *peer* interage com o mesmo módulo de manutenção da consistência. Este fato pode ser explorado para a integração de diferentes serviços de gerenciamento. Por exemplo, um módulo de processamento de políticas, um módulo de gerenciamento de falhas e um módulo de gerenciamento de configurações (possivelmente utilizando diferentes linguagens para representação interna dos dados de gerenciamento) poderiam ser integrados pelo módulo de manutenção da consistência por meio de justificativas.

Os serviços do módulo de manutenção de consistência são oferecidos através de uma interface simples que os módulos de serviços de gerenciamentos devem utilizar. O número de operações suportadas é restrito para facilitar a implementação nos módulos de serviços de gerenciamento. Além disso, detalhes das estratégias de comunicação não são especificados nas operações suportadas, dessa forma, mudanças nessas estratégias não levam a adaptações nas interfaces dos módulos de serviços de gerenciamentos. Assim, os métodos utilizados para executar os procedimentos relativos a mudanças de crenças são apresentados na Tabela 3.1.

Os métodos utilizados para executar os procedimentos descritos relativos a mudanças de crenças são um subconjunto dos métodos fornecidos pelo módulo de manutenção de consistência. Como pode ser visto na Tabela 3.1, dois métodos são utilizados para informar ao módulo de manutenção de consistência a presença ou ausência de justificativas ("justification_presence" e "justification_absence") e outros dois métodos são utilizados para consultar o estado de dados e a presença de justificativas ("datum_query" e "justification_query").

Os serviços fornecidos pelos módulos de serviços de gerenciamento e pelo módulo de manutenção de consistência necessitam ser disponibilizados localmente por meio de

alguma forma de comunicação interprocesso. Esses serviços podem ser disponibilizados de diversas formas, tais como, por exemplo, por meio de um barramento de mensagens como o *D-Bus* (*Desktop Bus*) (PENNINGTON, 2010) ou de um protocolo de comunicação interprocesso como o *DCOP* (*Desktop COmmunication Protocol* (MOORE, 2010).

O módulo de manutenção de consistência é concebido considerando que o *overlay* utilizado no sistema ANM baseado em P2P ofereça serviços de comunicação em grupo, como, por exemplo, o "PeerGroup Service" do *framework* JXTA (GONG, 2001). Uma possibilidade interessante como suporte para esse serviços é a utilização de *multicast*, mas não existe uma imposição sobre como o *overlay* P2P deva implementar esse suporte.

O *overlay* P2P deve prover suporte à organização de grupos, através dos serviços de gerenciamento disponibilizados. Dessa forma, o módulo de manutenção da consistência oferece um ambiente aberto para a integração rápida e dinâmica de módulos de serviços de gerenciamento de diferentes *peers*. Isso é possível em função de cada um dos *peers* executar seu próprio módulo de manutenção de consistência, o qual mantém sua própria estrutura de dados e controla a troca de mensagens dentro do grupo de *peers*. Dessa forma, módulos de serviços de gerenciamento se agrupam sem uma autoridade central, através de grupos de *peers*. Esses agrupamentos possuem características análogas com *federações*, conceito oriundo da literatura de MAS (HORLING; LESSER, 2005).

3.3 Estratégias de Comunicação

A proposta de manutenção de consistência apresentada na presente dissertação necessita que mudanças nas crenças nos *peers* sejam compartilhadas dentro de grupos de *peers* que compõem um sistema ANM baseado em P2P. O módulo de manutenção de consistência controla a troca de mensagens por meio de serviços de comunicação do *overlay* P2P. Nessa troca de mensagens, mudanças nas crenças são adaptadas em mensagens a serem difundidas no grupo de *peers* e vice-versa. O modelo utilizado para o *overlay* P2P é não estruturado, assim, não existe relação entre as informações armazenadas em um *peer* e sua posição na topologia do *overlay*. Além disso, *peers* não têm acesso ao tamanho do grupo de *peers*, assim como a identidades dos *peers* participantes.

A operação de uma infraestrutura fracamente acoplada (como um *overlay* P2P não estruturado) dificulta a utilização de estratégias convencionais como troca explícita de mensagens e utilização de memória compartilhada para coordenação (DIMOPOULOS; MORAITIS, 2006). Dessa forma, as estratégias de comunicação utilizadas devem estar preparadas para enfrentar diversos desafios. Um deles é a dinamicidade do conjunto de *peers*, já que os mesmos podem entrar e sair do grupo durante sua operação. Além disso, a lista de identidades dos *peers* não é conhecida pelos módulos de serviços de gerenciamento e pelo módulo de manutenção da consistência, assim, essa identidade não pode ser utilizada na troca de mensagens. É importante ressaltar também que não são utilizados mecanismos de reconhecimento fim a fim, dessa forma, mensagens podem não chegar a todos os *peers* de um grupo de *peers* (*i.e.*, não existe confiabilidade na troca de mensagens).

Agregações são uma abstração interessante para prover escalabilidade para sistemas distribuídos (VAN RENESSE; BIRMAN; VOGELS, 2003). Na presente proposta é utilizada a premissa que existe suporte à organização de grupos de *peers* por meio de módulos de serviços de gerenciamento. Além disso, *peers* podem participar de diversos grupos de acordo com os módulos que possuem. A utilização de grupos de *peers* é fundamental para a escalabilidade dessa proposta, já que a troca de mensagens é restrita a cada grupo de

peers. Outros mecanismos para consistência de informações compartilhadas em sistemas distribuídos utilizam conceitos similares, como, por exemplo, "*domains*", um conceito utilizado no *Astrolabe* (VAN RENESSE; BIRMAN; VOGELS, 2003).

Os métodos utilizados para a troca de mensagens dentro do grupo de *peers* são modelados utilizando-se conceitos oriundos de modelos de computação distribuída inspirados na Biologia (BABAOGLU et al., 2006). Entre esses, modelos baseados em *proliferação* são uma alternativa interessante para os requisitos de comunicação da presente proposta. Neste contexto, uma mudança de crença possivelmente passa por proliferação nos *peers* visitados, em que cada *peer* calcula a probabilidade de encaminhar a mudança de crença usando uma função de controle de proliferação. Todos os *peers* de um grupo de *peers* executam exatamente a mesma função de controle de proliferação, que podem ser iniciadas por qualquer um dos *peers* do grupo de *peers*.

Diversos modelos baseados em proliferação foram descritos na literatura (BABAOGLU et al., 2006). Dentre esses modelos, a presente proposta se apóia em características encontradas na *replicação*. Esse modelo pode oferecer suporte para diferentes estratégias de comunicação. Na presente proposta, a replicação é representada por meio da difusão de mensagens pelos *peers* para replicar mudanças em crenças entre as entidades participantes (*i.e.*, *peers* de um grupo de *peers* específico). Processos eficientes e satisfatórios baseados em replicação são comuns na natureza (*e.g.*, processos de crescimento, espalhamento epidêmico, processos de proliferação no sistema imunológico) (BABAOGLU et al., 2006).

As estratégias de comunicação utilizadas estão em acordo com o modelo de consistência empregado nesta dissertação. Nesse modelo, um *peer* irá refletir as atualizações recebidas de forma não determinística por outros *peers*; dessa forma a consistência só poderá ser alcançada após transcorrer tempo suficiente para que as mensagens sejam difundidas dentro do grupo de *peers*. Os mecanismos de replicação estão integrados a esse modelo de consistência, frente às peculiaridades desses mecanismos.

Duas formas de replicação foram desenvolvidas na presente dissertação: a replicação livre e a replicação controlada. Essas formas são descritas a seguir.

3.3.1 Replicação Livre

A replicação, em uma forma abstrata, é definida nesta proposta como o processo onde os *peers* recebem mudanças de crença por meio de mensagens do grupo e encaminham algumas das mudanças de crenças recebidas de acordo com regras específicas da aplicação. Neste contexto, a replicação livre (NOBRE; GRANVILLE, 2009a) (NOBRE; GRANVILLE, 2009b) é o processo em que todas as mudanças de crença são replicadas entre as entidades participantes (*i.e.*, *peers* de um grupo de *peers* específico). Além disso, as mudanças de crenças são replicadas imediatamente após ocorrerem. Após o recebimento de uma mudança, os *peers* verificam se a mesma representa uma novidade e atualizam suas crenças.

A replicação livre é implementada pela inundação de mudanças em crenças dentro do grupo de *peers*. Técnicas relacionadas com a inundação são usualmente utilizadas, por exemplo, para implementar operações de busca em redes *overlay* não estruturadas. A inundação proporciona características de robustez ao sistema e alta velocidade na atualização dos diferentes *peers* do *overlay* P2P. No entanto, a inundação produz um número muito grande de mensagens, aumentando o consumo de banda por tráfego de gerenciamento e a quantidade de recursos utilizados para o processamento de mensagens.

A escalabilidade da replicação livre está diretamente relacionada com a definição dos

grupos de *peers*. Dentro deles, a manipulação das mensagens inundadas é realizada somente pelo descarte de mensagens recebidas. Dessa forma, um *peer* pode difundir uma mudança de crença por meio de uma mensagem, apesar de ter recebido diversas mensagens iguais informando a mesma mudança de crença.

Melhorias podem ser realizadas para refinar as regras de transmissão de mensagens, incluindo a utilização de informações relacionadas ao comportamento temporal das comunicações. Essa utilização poderia permitir a criação de grupos de *peers* com grande número de membros. Apesar de grupos de *peers* com essas características não serem comuns em sistemas ANM baseados em P2P, a utilização de um grande número de *peers* pode trazer novas possibilidades ao gerenciamento de redes distribuído.

3.3.2 Replicação Controlada

A replicação controlada foi desenvolvida em função da verificação das limitações da replicação livre para grupos com grande número de *peers* (NOBRE; GRANVILLE, 2010). O objetivo é utilizar um número significativamente inferior de mensagens em mudanças de crença do que na replicação livre. O encaminhamento de mensagens por meio da replicação controlada significa que algumas mudanças de crenças não serão encaminhadas devido a regras de restrição. Esse controle é realizado para minimizar utilizações redundantes da infraestrutura de comunicações.

A replicação é controlada por uma *Função de Controle da Replicação* (*Replication Controlling Function* - RCF). O RCF define a probabilidade '*P*' de uma mudança de crença ser encaminhada para o grupo de *peers*. Em termos gerais, essa probabilidade poderia ser definida utilizando-se diferentes parâmetros da operação do *overlay* de gerenciamento. Inicialmente, a RCF foi definida como demonstrado na Equação 3.1:

$$P(\rho, \eta b) = \frac{\rho}{1 + \eta b} \tag{3.1}$$

em que ηb representa o número de mensagens recebidas de uma mudança de crença específica dentro de um "backoff delay" (*T*) e ρ representa uma constante de proliferação positiva. A essência dessa função é que a proliferação de uma mudança de crença deve diminuir com o recebimento de múltiplas cópias de uma mudança de crença (*i.e.*, de sua mensagem). Além disso, a Equação 3.1 assegura que $0 < P < \rho$ (para ηb contido no conjunto dos número naturais). A equação é avaliada no tempo *T*.

Peers enviam mudanças de crença produzidas internamente com $P = 1$ para o grupo de *peers* (*i.e.*, a mudança de crença é sempre enviada). Quando a mudança de crença é recebida pelos serviços de comunicação do *overlay* P2P, a recepção de novas mensagens informando a mudança de crença já recebida durante *T* (*backoff delay*) diminui a probabilidade dessa mudança de crença ser encaminhada para o grupo de *peers*. Pode-se ressaltar que quando $\rho = 1$ e $T = 0$ ou $\eta b = 0$, a replicação controlada tem o mesmo comportamento da replicação livre. Dessa forma, pode-se afirmar que a replicação livre é um caso especial da replicação controlada.

Os parâmetros da RCF podem ser otimizados de acordo com características encontradas no processo de replicação, como propriedades dos *peers* e particularidades dos dados compartilhados no grupo de *peers*. Essa otimização pode ser realizada de forma estática, por meio de ensaios para verificar melhores valores, ou dinâmica, por exemplo, por meio de técnicas de Inteligência Artificial Distribuída como Aprendizagem por Reforço (*Reinforcement Learning* - RL) Multiagente. Também podem ser utilizados esquemas de compressão, condensando diversas mudanças de crença em apenas uma mensagem e/ou

realizando atualizações periódicas. As possibilidades de otimização descritas se configuram como uma área para trabalhos futuros relacionados com a presente proposta.

4 AVALIAÇÃO

A utilização de *overlays* P2P como infraestrutura para sistemas de Gerenciamento Autonômico de Redes (Autonomic Network Management - ANM) descentralizados - a qual produz sistemas ANM baseados em P2P - traz preocupações adicionais sobre a efetiva avaliação desses sistemas. É preciso compreender e estimar adequadamente quais são os *trade-offs* dessa utilização. Por exemplo, potenciais dificuldades em aspectos temporais e na utilização eficiente de recursos devem ser acompanhadas, como contrapartida, de melhorias em confiabilidade e responsividade em operação permanente.

O restante do capítulo é organizado da seguinte forma. Na seção 4.1 são apresentadas métricas utilizadas na presente avaliação. Na seção 4.2 é apresentada uma avaliação qualitativa entre a proposta descrita no capítulo 3 e algumas alternativas para manutenção da consistência do estado dos dados de gerenciamento em sistemas ANM baseados em P2P. Na seção 4.3 são apresentados dois estudos de caso: o gerenciamento de falhas em enlaces de rede Ethernet em provedores de serviços e a manutenção da consistência do estado de ativação de políticas de gerenciamento de rede. Na seção 4.4 são apresentados experimentos simulados para a avaliação das estratégias de comunicação descritas no capítulo 3.

4.1 Métricas de Avaliação

Alguns trabalhos (MCCANN; HUEBSCHER, 2004) (BABAOGLU et al., 2006) enumeram métricas que devem ser consideradas na avaliação de sistemas autonômicos, como QoS, custo, granularidade/flexibilidade, escalabilidade, robustez, grau de autonomia, adaptatividade, insensibilidade, estabilização e *benchmarkings*. Outra interessante fonte de métricas é o estudo comparativo de trabalhos realizados em outras áreas da computação autonômica (MCCANN; HUEBSCHER, 2004). Contudo, apenas algumas dessas métricas serão discutidas, consideradas mais importantes pela presente avaliação, de acordo com especificidades de sistemas ANM baseados em P2P.

Escalabilidade é uma das métricas mais importantes na avaliação de sistemas distribuídos. Dessa forma, essa métrica deve ser observada na utilização de *overlays* P2P como infraestrutura de diferentes sistemas, por exemplo, pela análise dos custos de comunicação intrínsecos ao *overlay*. Além disso, mecanismos adicionais implementados nesses *overlays* (*e.g.*, mecanismos para a manutenção de consistência) também devem ser escaláveis. Apesar da escalabilidade não apresentar muitos obstáculos para ser medida, a mesma é correlacionada com outras métricas, as quais precisam ser controladas para uma avaliação justa (BABAOGLU et al., 2006).

Robustez é outra métrica bastante importante na avaliação de sistemas distribuídos. Sistemas baseados em centralização são mais vulneráveis a falhas e ataques (VAN RE-

NESSE; BIRMAN; VOGELS, 2003). Alguns autores apontam que a utilização de *overlays* P2P como infraestrutura para sistemas de gerenciamento de redes apresenta boas propriedades em relação a essa métrica (GRANVILLE et al., 2005). Além disso, alguns trabalhos demonstram que essas propriedades também são verificadas em sistemas ANM baseados em P2P (MARQUEZAN et al., 2007).

Grau de autonomia é uma métrica utilizada para mostrar o quão autônomo um sistema é (MCCANN; HUEBSCHER, 2004). Em sistemas distribuídos, essa métrica pode ser também utilizada para avaliar a autonomia de cada um dos elementos que formam esses sistemas. Os *peers* do *overlay* P2P são os elementos a serem avaliados em um sistema ANM baseado em P2P. No entanto, aumentos no grau de autonomia podem produzir dificuldades em outras métricas, como, por exemplo, estabilização.

Estabilização é uma métrica que quantifica a estabilidade da operação de um sistema específico ou dos elementos que o formam. Essa métrica é avaliada por meio da relação entre mudanças percebidas no ambiente e as respectivas respostas produzidas pelo sistema. A introdução de automatização e distribuição em um sistema pode dificultar a estabilidade do mesmo, dessa forma, essa métrica é importante na avaliação de sistemas ANM baseados em P2P

4.2 Comparação

Existem algumas alternativas para manter a consistência do estado de dados de gerenciamento em um sistema ANM baseado em P2P. O capítulo 3 apresenta a alternativa descrita na presente dissertação: a introdução de funcionalidades de manutenção da verdade multiagente. Duas alternativas adicionais serão também descritas para a realização de uma avaliação qualitativa: a utilização de um repositório compartilhado e a passagem de bastão. Essas alternativas adicionais foram escolhidas por razões de simplicidade, tornando essa avaliação mais intuitiva. As métricas utilizadas nessa avaliação foram descritas na seção 4.1.

Sistemas ANM baseados em P2P utilizam os serviços de comunicação do *overlay* P2P para a difusão de mensagens entre os *peers*. Esse *overlay* pode ser modelado como uma rede de comunicação assíncrona não confiável, assim, devem ser consideradas questões como atrasos e perdas de mensagens. Em função dessas questões, sistemas ANM baseados em P2P podem ser descritos como sistemas distribuídos assíncronos. Além disso, são bem conhecidas as dificuldades que os sistemas distribuídos assíncronos impõem para alcançar consenso ou acordo (FISCHER; LYNCH; MERRITT, 1990). Dessa forma, as alternativas apresentadas procuram enfrentar essas dificuldades e prover suporte à manutenção de consistência, sendo que cada alternativa possui suas particularidades.

A mudança no estado de um dado de gerenciamento devido à recepção de um sinal assíncrono é utilizada como exemplo para uma avaliação comparativa. Nesse exemplo, um sinal assíncrono é recebido pelo sistema ANM baseado em P2P, e a mudança ocasionada pela recepção - novo estado do dado de gerenciamento - deve ser armazenada de forma consistente na Base de Conhecimento (*Knowledge Base* - KB) desse sistema. A atualização dessa KB pode ser realizada de diversas formas, de acordo com cada alternativa de manutenção da consistência do estado dos dados de gerenciamento em sistemas ANM baseados em P2P apresentada. Uma avaliação para as alternativas apresentadas em relação a esse exemplo é apresentada na Tabela 4.1.

A utilização de **funcionalidades de manutenção da verdade multiagente** provê manutenção da consistência do estado dos dados de gerenciamento em sistemas ANM base-

ados em P2P por meio da troca de crenças sobre os dados de gerenciamento entre os *peers* de um sistema ANM baseado em P2P. Essas crenças são representadas por justificativas e suas presenças, conforme apresentado no capítulo 3. Após a recepção de um sinal assíncrono, os *peers* enviam uma mensagem para o grupo de *peers* informando mudanças nas presenças de justificativas. Quando recebem essa mensagem, os outros *peers* verificam se essa mudança é condizente com sua KB, atualizando-a, se necessário.

A utilização de funcionalidades de manutenção da verdade multiagente não modifica as propriedades de descentralização de um sistema ANM baseado em P2P, mantendo cada *peer* como uma entidade independente e autônoma. Como muitas avaliações já demonstraram, um sistema que não compartilhe elementos possui boas características de escalabilidade. Além disso, mantendo a autonomia de cada *peer*, Pontos Únicos de Defeito (*Single Points of Failure* - SPoF) são eliminados. No entanto, o grau de autonomia de cada *peer* é diretamente relacionado à predição global do sistema ANM baseado em P2P; dessa forma, podem haver dificuldades frente à estabilidade do sistema. As avaliações qualitativas para essa alternativa são apresentadas na Tabela 4.1.

Uma das alternativas para a manutenção da consistência do estado dos dados de gerenciamento em sistemas ANM baseados em P2P é a utilização de um **repositório compartilhado** para armazenar o estado dos dados de gerenciamento. Assim, esses estados são armazenados de forma centralizada, o que pode ser realizado em um *peer* especial (*i.e.*, *super peer*) ou mesmo em um servidor normal (*i.e.*, qualquer ponto único de armazenamento). Centralizações podem ser vistas em alguns sistemas P2P pela utilização de um "servidor central" (*e.g.*, Napster). Após a recepção de um sinal assíncrono, os *peers* enviam uma mensagem ao servidor para informar uma modificação em dados de gerenciamento. Após receber essas mensagens, o servidor atualiza sua KB e envia uma mensagem para o grupo de *peers*, os quais atualizam suas KBs (se necessário).

A utilização de um repositório compartilhado possui características interessantes em relação à estabilidade do sistema, já que o único estado "oficial" dos dados de gerenciamento é aquele registrado nesse repositório. No entanto, a utilização compartilhada de um repositório centralizado levanta questões em relação à escalabilidade e robustez, já que o servidor é um SPoF e um gargalo em relação a recursos computacionais e de comunicação. Além disso, essa centralização diminui o grau de autonomia dos *peers* que formam sistemas ANM baseados em P2P. As avaliações qualitativas para essa alternativa são apresentadas na Tabela 4.1.

A segunda alternativa comparada para a manutenção da consistência do estado de gerenciamento em sistemas ANM baseados em P2P é a utilização de **passagem de bastão** para eleger um *peer* como "ativo" no grupo de *peers*. Nessa alternativa, quando o *peer* ativo passa o bastão para outro *peer* (*i.e.*, o próximo *peer* ativo), também são enviados os estados dos dados de gerenciamento para o próximo *peer* ativo. Esse esquema é utilizado em alguns sistemas P2P como uma ferramenta para tarefas operacionais, como, por exemplo, contabilização (LIEBAU et al., 2005). Após a recepção de um sinal assíncrono, os *peers* apenas atualizam sua própria base de conhecimento, consultando o *peer* ativo para a execução de tarefas de gerenciamento. Na passagem do bastão, o próximo *peer* ativo verifica se seus estados são coerentes com sua KB e, em caso negativo, atualiza-a.

A utilização de passagem de bastão possui características interessantes em relação à estabilidade do sistema, já que o único estado "oficial" dos dados de gerenciamento é aquele registrado no *peer* ativo. Contudo, a operação de troca de bastão é bastante crítica em relação à robustez e escalabilidade, o que, em geral, implica em procedimentos adicionais para solucionar problemas relativos à perda do bastão ou a duplicação do

	Métricas			
Esquemas	Escalabilidade	Robustez	Autonomia	Estabilização
Manutenção da Verdade Multiagente	forte	forte	forte	fraco
Repositório Compartilhado	fraco	fraco	fraco	forte
Passagem de Bastão	fraco	fraco	fraco	forte

Tabela 4.1: Avaliação comparativa de alternativas para a manutenção da consistência do estado dos dados de gerenciamento

mesmo. Além disso, como existe apenas um *peer* ativo em um dado momento, a utilização de passagem de bastão diminui o grau de autonomia dos *peers* que formam sistemas ANM baseados em P2P. As avaliações qualitativas para essa alternativa são apresentadas na Tabela 4.1.

O estudo comparativo de alternativas para a manutenção da consistência do estado de gerenciamento em sistemas ANM baseados em P2P mostram alguns pontos fortes e fracos de cada uma das alternativas. No entanto, algoritmos específicos utilizados em diferentes implementações de cada alternativa podem influenciar seu desempenho nas métricas citadas. Apesar deste fato, existe um bom acordo em muitos casos, o que possibilita a avaliação qualitativa e comparativa.

4.3 Estudos de Caso

Estudos de caso podem ser utilizados para avaliar a utilização de funcionalidades de manutenção da verdade multiagente para a manutenção de consistência do estado dos dados de gerenciamento em um sistema ANM baseado em P2P. Esses estudos podem ser utilizados para apresentar as características da proposta descrita na presente dissertação e demonstrar a viabilidade de sua implementação em contextos operacionais.

A manutenção da consistência do estado dos dados de gerenciamento é importante em diversas tarefas de gerenciamento de redes em um sistemas ANM baseado em P2P. *Peers* devem estar habilitados a compartilhar o estado desses dados para uma execução eficiente dessas tarefas e para o sistema ANM baseado em P2P apresentar um comportamento coerente. Assim, estudos de caso podem ser construídos a partir da descrição de tarefas de gerenciamento, com ênfase naquelas em que a consistência do estado dos dados de gerenciamento é preponderante.

Estudos de caso também podem demonstrar se a proposta descrita na presente dissertação pode ser generalizada de sistemas ANM baseados em P2P para outros sistemas ANM descentralizados. Conforme descrito no capítulo 3, os elementos constitutivos dos sistemas ANM baseados em P2P (*i.e.*, *peers* e grupos de *peers*) podem ser abstraídos para os elementos constitutivos dos sistemas ANM descentralizados (*i.e.*, AMEs e AMDs). Assim, a utilização de manutenção da verdade multiagente para a manutenção de consistência do estado dos dados de gerenciamento pode ser também aplicável para sistemas ANM descentralizados em geral.

O restante da seção apresenta dois estudos de caso, a fim de avaliar a proposta descrita

na presente dissertação. O primeiro estudo de caso descreve o gerenciamento de falhas em enlaces de rede Ethernet em provedores de serviços. O segundo estudo de caso descreve a manutenção da consistência do estado de políticas.

4.3.1 Gerenciamento de Falhas em enlaces de rede Ethernet em Provedores de Serviços

O suporte à colaboração para a realização de tarefas de gerenciamento é uma das novas demandas enfrentadas por sistemas de gerenciamento de redes. A borda de provedores e consumidores de serviços de rede é um exemplo de ambiente que impõe desafios para a realização de tarefas colaborativas de gerenciamento. Uma das tarefas mais importantes nesse ambiente é a detecção de falhas em enlaces que conectam o provedor de serviços ao consumidor de serviços. É desejável o compartilhamento de responsabilidades sobre essa detecção, entre o pessoal responsável em ambos domínios (provedor e consumidor de serviços). Além disso, algum grau de automação é necessário para realizar efetivamente essas tarefas colaborativas (PRAS et al., 2007), o qual deve ser provido pelo sistema de gerenciamento de redes.

O gerenciamento de falhas de um enlace que conecta dois domínios diferentes (*e.g.*, provedor de serviços e consumidor de serviços) precisa de informações de diferentes fontes, como dispositivos gerenciados e administradores de rede. Em cada domínio, o pessoal responsável manipula a detecção de notificações de falhas enviadas por dispositivos para gerar conhecimento humano sobre o estado do enlace. Dessa forma, são compartilhadas responsabilidades entre os administradores de rede dos dois domínios sobre a detecção de falhas.

A integração das informações sobre notificações de falhas, em conjunto com conhecimento humano, produz dados de gerenciamento, os quais podem assumir diferentes estados. Esses dados podem ser utilizados em diversas tarefas de gerenciamento. Podem, por exemplo, ser utilizados para confrontar Acordos de Nível de Serviço (*Service Level Agreements* - SLAs), por ambos domínios, a fim de fundamentar ou esclarecer infrações a níveis de serviço. A fim de facilitar tarefas de gerenciamentos como a citada, esses dados devem ser mantidos pelo sistema de gerenciamento de redes de uma forma distribuída e autônoma, mantendo sua integridade e consistência. No entanto, na maior parte dos casos, esses dados estão disponíveis apenas em uma entidade única ou em um pequeno conjunto de entidades hierárquicas.

A utilização de Ethernet em redes metropolitanas é uma das possibilidades mais interessantes dentre as tecnologias atualmente utilizadas para conectar provedores e consumidores de serviços (*i.e.*, redes de acesso). Dessa forma, essa tecnologia foi utilizada para a construção deste estudo de caso. Além disso, alguns autores declaram que a utilização de Ethernet em provedores de serviços pode qualificar muitos aspectos relativos à interconectividade e ao investimento dos consumidores (SANCHEZ; RAPTIS; VAXEVANAKIS, 2008). Neste contexto, cada instância de serviço de consumidor deve ser gerenciada individualmente pelos dois domínios (provedor e consumidor de serviços).

Uma das principais funcionalidades das ferramentas “Operações, Administração e Manutenção” (*Operations, administration, and maintenance* - OAM) do Ethernet é o gerenciamento de falhas (MCFARLAND; SALAM; CHECKER, 2005). O suporte aos requisitos de SLA na borda provedor/consumidor de serviços necessita a utilização de ferramentas OAM (RYOO et al., 2008). Os serviços oferecidos necessitam ser gerenciados com a granularidade de uma *instância de serviço de consumidor*. Essa instância é definida como uma Conexão Virtual Ethernet (*Ethernet Virtual Connection* - EVC), de

acordo com a nomenclatura utilizada pelo *Metro Ethernet Forum* (MEF) (MCFARLAND; SALAM; CHECKER, 2005).

Um dos mecanismos disponibilizados nas ferramentas OAM do Ethernet para prover suporte ao gerenciamento de falhas são as mensagens "*Alarm Indication Signal*" (AIS) (MCFARLAND; SALAM; CHECKER, 2005). Essas mensagens são disparadas quando ocorre alguma falha entre dois nós, para ambas direções. Assim, mensagens AIS possibilitam notificações assíncronas para outros elementos na rede de que existe uma falha na rede Ethernet (SANCHEZ; RAPTIS; VAXEVANAKIS, 2008). Essas mensagens são enviadas periodicamente para assegurar que, enquanto a falha persiste, esse estado será mantido (MCFARLAND; SALAM; CHECKER, 2005).

Um provedor de serviços Ethernet poderia utilizar traps SNMP (HARRINGTON; PRESUHN; WIJNEN, 2002) para verificar a ocorrência de falhas. No entanto, quando uma falha ocorre, o provedor de serviços pode não ter noção sobre quais consumidores serão impactados. Utilizando as mensagens AIS, o mecanismo de continuidade determina quais as EVCs serão impactados. Dessa forma, o provedor de serviço saberá exatamente quais serviços dos consumidores estão com problemas. Evidentemente, os esforços para gerenciar serviços Ethernet (Camada de Enlace de Dados) devem considerar uma infraestrutura sobreposta, em geral constituída utilizando-se tecnologia IP (RYOO et al., 2008).

O reconhecimento humano das falhas em enlaces de rede auxilia na produção de dados de gerenciamento que integram aspectos operacionais automatizados e o conhecimento dos administradores de rede. Assim, a detecção de falhas também deve considerar conhecimento humano, advindos dos administradores de rede de ambos domínios (provedor de serviços e consumidor de serviços). Uma aplicação particularmente importante para esses dados é a avaliação de infrações a SLAs na borda provedor/consumidor de serviços. Essa importância é verificada, por exemplo, porque esses dados podem ser utilizados para assegurar que essas avaliações sejam realizadas fim a fim.

Este estudo de caso apresenta um exemplo de detecção de uma falha em um enlace de rede por meio de mensagens AIS e conhecimento humano. Assim, dentro do sistema ANM baseado em P2P, conforme explicado no capítulo 3, os *peers* formam grupos de acordo com os serviços de gerenciamentos disponibilizados. Nesse exemplos, os *peers* que formam o grupo de *peers* possuem um módulo de serviço de gerenciamento que coleta mensagens AIS e outro módulo que coleta informações de administradores humanos. Além disso, há o módulo de manutenção de consistência, responsável por integrar esses módulos de serviço de gerenciamento e manter a consistência do estado dos dados de gerenciamento. A Figura 4.1 demonstra uma borda provedor/consumidor de serviços conectada por uma EVC e um grupo de *peers* responsável pelo gerenciamento de falhas nessa conexão.

O dado que representa a detecção de uma falha de enlace (*Link Fault Detected*), é justificado por comandos de administradores humanos em ambos domínios, Provedor de Serviços (*Service Provider Detection*) e Consumidor de Serviços (*Service Consumer Detection*), e pelo recebimento de mensagens AIS (*Device Notification Received*). O módulo de serviço de gerenciamento *coletor de notificação de dispositivos* é responsável por coletar informações sobre mensagens AIS. Da mesma forma, o módulo de serviço de gerenciamento *coletor de detecção humana* é responsável por coletar o comando de detecção efetuado pelos administradores humanos. A Figura 4.2 mostra a relação entre o módulo de manutenção da consistência e os módulos de serviços de gerenciamento utilizados neste estudo de caso.

O módulo de manutenção de consistência recebe o dado "*Link Fault Detected*"

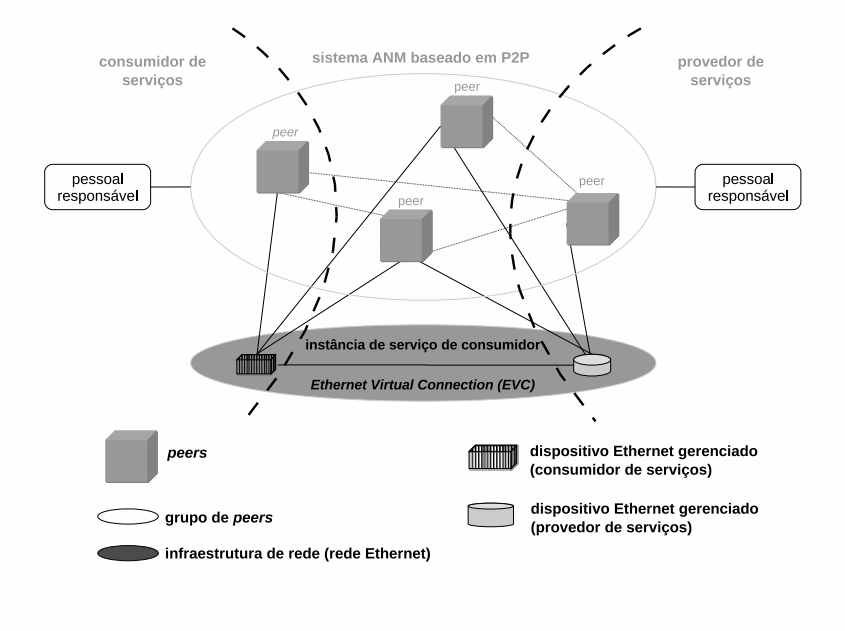

Figura 4.1: Borda entre consumidor e provedor de serviços gerenciada por um sistema ANM baseado em P2P

(link_flt_det) e sua lista de justificativas, composta por "*Service Operator Detection*" (srv_prv_det), "*Service Consumer Detection*" (svr_con_det) e "*Device Notification Received*" (dev_not_rcv). A definição do dado e de sua lista de justificativas é realizada por um administrador humano ou por um sistema especialista, conforme descrito no capítulo 3. O trecho de código abaixo apresenta a representação interna aos *peers* desse dado e de sua lista de justificativas. Essa representação deve ser compartilhada entre todos os *peers* que formam o grupo de *peers* relativo a esse dado dentro do sistema ANM baseado em P2P.

```
justification(srv_prv_det).
justification(srv_con_det).
justification(dev_not_rcv).
justificationIsPresent(X) :- generated(X).
justificationIsPresent(X) :- received(X).
datum(link_flt_det) :-
 justificationIsPresent(srv_prv_det),
 justificationIsPresent(srv_con_det),
 justificationIsPresent(dev_not_rcv).
datumIsInternal(link_flt_det) :-
 generated(srv_prv_det),
 generated(srv_con_det),
 generated(dev_not_rcv).
```

A presença das justificativas pode ser definida pelos módulos de serviço de gerenciamento coletor de notificação de dispositivos e coletor de detecção humana (gerada internamente) ou pelo módulo de manutenção de consistência (recebida por mensagens no grupo de *peers*). As regras "justificationIsPresent" representam as possibilidades para definição da presença das justificativas por meio da utilização dos fatos "generated", para presenças geradas internamente, e "received", para presenças recebidas por meio de mensagens no grupo de *peers*.

Uma situação pode ser apresentada como exemplo deste estudo de caso: devido a problemas na recepção de uma mensagem AIS, alguns *peers* do grupo de *peers* entram em

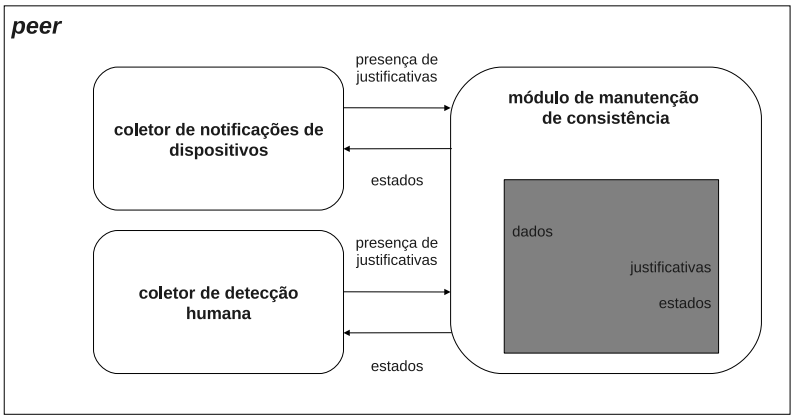

Figura 4.2: Arquitetura dos *peers* para o gerenciamento de falhas em enlaces de rede Ethernet em provedores de serviços

desacordo sobre a ocorrência de falha em um enlace de dados (*i.e.*, uma EVC). Apesar desse problema, comandos executados por administradores de rede localizados no provedor de serviços e no consumidor de serviços detectaram a falha e foram recebidos por todos os *peers* do grupo de *peers*. Essa situação pode tornar o grupo de *peers* inconsistente.

A proposta descrita na presente dissertação define que módulos de serviço de gerenciamento informam mudanças de crença para o módulo de manutenção de consistência, o qual difunde essas mudanças dentro do grupo de *peers* conforme descrito no capítulo 3. Essa difusão poderia ser utilizada para evitar que o não recebimento de uma mensagem AIS por algum dos *peers* torne o grupo de *peers* inconsistente. Assim, mesmo um *peer* que não produziu a presença de alguma de suas justificativas, mas recebeu uma mensagem informando sobre a mudança nessa presença, está coerente com outros *peers* do grupo de *peers*. O código abaixo demonstra a resposta produzida pelo módulo de manutenção de consistência nesta situação.

```
link_flt_det:external (srv_prv_det:mod srv_con_det:mod dev_not_rcv:msg)
```

A resposta produzida pelo módulo de manutenção de consistência indica que o estado do dado de gerenciamento "*Link Fault Detected*" (link_flt_det) é "in" e marcado com a opção "external". Além disso, pode ser percebido que a presença da justificativa "*Device Notification Received*" (dev_not_rcv) foi verificada por meio do recebimento de mensagens de mudanças de crenças (indicada pelo ":msg" no trecho de código). Dessa forma, o estado da dado de gerenciamento se mantém consistente dentro do grupo de *peers*, e o sistema ANM baseado em P2P apresenta um comportamento coerente.

As ferramentas de OAM do Ethernet ainda possibilitam a utilização de uma sinalização destinada a diferenciar uma condição de falha e um bloqueio intencional administrativo na EVC (*e.g.*, para diagnóstico). Essa sinalização é realizada pela mensagem "*Locked Signal Message* (LCK) (RYOO et al., 2008). Conforme a presente proposta, a possibilidade de gerenciar essa sinalização pode se realizar pela introdução de uma nova

justificava "*Device Diagnostics Off*" (dev_dia_off), de acordo com o trecho de código abaixo.

```
justification(dev_dia_off).
```

A lista de justificativas do datum "*Link Fault Detected*" precisa ser modificada para refletir a introdução da nova justificativa "*Device Diagnostics Off*" (dev_dia_off). Essa modificação é descrita no trecho de código abaixo. Assim, pode-se verificar a facilidade na introdução de novas informações para a realização de uma tarefa de gerenciamento por meio da utilização de justificativas.

```
datum(link_flt_det) :-
 justificationIsPresent(srv_prv_det),
 justificationIsPresent(srv_con_det),
 justificationIsPresent(dev_not_rcv),
 justificationIsPresent(dev_dia_off).
datumIsInternal(link_flt_det) :-
 generated(srv_prv_det),
 generated(srv_con_det),
 generated(dev_not_rcv)
 generated(dev_dia_off).
```

A justificativa "*Device Diagnostics Off*" (dev_dia_off) define a ausência de mensagens LCK, demonstrando que não está sendo realizado um bloqueio administrativo do enlace de rede gerenciado. A utilização de justificativas que representam ausências substitui a funcionalidade de listas de suporte de justificativas conforme descrito no capítulo 3. Dessa forma, a representação de uma ausência (*i.e.*, crença não acreditada) é definida em manutenção da verdade multiagente como parte de uma lista "OUT".

O gerenciamento de falhas de enlaces de dados é uma tarefa de gerenciamento tradicionalmente realizada por meio de sistemas centralizados sem integração. Em geral, esses sistemas são: um sistema de gerenciamento de redes (que coleta notificações dos dispositivos); e um *Trouble Ticket System* (TTS) (que coleta informações oriundas de administradores humanos). O procedimento tradicional apresenta problemas em escalabilidade e robustez relativos à utilização de centralização, já que cada um dos sistemas (sistema de gerenciamento de redes e TTS) é um ponto único de defeito (*Single Point of Failure - SPoF*) e também um gargalo em termos de desempenho.

O procedimento tradicional para o gerenciamento de falhas de um enlace de dados ainda impõe dificuldades na integração das informações entre os diferentes sistemas. Esses sistemas podem operar internamente com modelos de dados e linguagens distintas. A proposta descrita na presente dissertação permite que essa integração seja realizada por meio da utilização de justificativas para dados de gerenciamento distribuído em grupos de *peers* de um sistema ANM baseado em P2P. Os dados de gerenciamento e suas justificativas são manipulados pelo módulo de manutenção de consistência, os quais podem ser utilizados para diferentes processos, como, por exemplo, auditoria e contabilização.

4.3.2 Estado de Ativação de Políticas

O presente estudo de caso é uma ilustração da manutenção de consistência do estado de políticas em um sistema ANM baseado em P2P. Políticas possuem um papel preponderante em sistemas ANM (EMANICS, 2010) (JENNINGS et al., 2007). O Gerenciamento de Redes baseado em Políticas (*Policy-Based Network Management* - PBMN) tem se mostrado uma abordagem interessante para o gerenciamento de redes (SLOMAN, 1994), além de representar uma tecnologia que provê suporte a capacidades de gerenciamento

autonômicas (EMANICS, 2010). Por exemplo, políticas podem ser utilizadas para automatizar processos de tomada de decisão.

Sistemas ANM baseado em P2P com suporte a políticas necessitam que seus elementos constitutivos - *peers* - estejam preparados para a utilização de políticas e, da mesma forma, participem do processo de tomada de decisão relativo às tarefas de gerenciamento. Essas decisões são processadas por meio da análise de requisições em relação a uma política específica. Após esse processo, uma decisão final é repassada ao elemento que a solicitou, o qual pode ser um dispositivo de rede ou um administrador humano.

Políticas de obrigação (*obligation policies*) são utilizadas como modelo neste estudo de caso. Uma política de obrigação é um objeto gerenciado que é instanciado e, por meio da recepção de um evento esperado e da avaliação de zero ou mais condições, uma ou mais ações são executadas (DAMIANOU et al., 2010). Dessa forma, políticas de obrigação são normalmente descritas na forma de regras Evento-Condição-Ação (*Event-Condition-Action* - ECA)

O estado da política (*i.e.*, ativação da política) é definido como um dado de gerenciamento, e eventos e condições são definidos como justificativas no módulo de manutenção de consistência, conforme apresentado no capítulo 3. Esse módulo provê um ponto de integração para possibilitar um comportamento global coerente de um sistema ANM baseado em P2P, em conjunto com a utilização de políticas.

As políticas são descritas neste estudo de caso utilizando-se *Ponder2* (DAMIANOU et al., 2010). *Ponder2* é um *toolkit* que oferece suporte à especificação e à aplicação de políticas em diversas formas, tais como, por exemplo, regras ECA. Quando a ocorrência de um evento é anunciada ao *Ponder2*, o componente “Obligation Policy Interpreter” (responsável pelo processamento de políticas) verifica essa ocorrência contra os eventos definidos nas políticas registradas e, em caso positivo, avalia as condições nas políticas relevantes. Se as condições também forem satisfeitas, as ações descritas nas políticas relevantes são executadas.

O trecho de código abaixo demonstra a codificação XML (*eXtensible Markup Language*) de uma política de obrigação que responderá a eventos do tipo */event/humanResourcesProcedureEvent*. Neste exemplo, após a recepção do evento citado, o *Ponder2* verifica se a carga em um roteador “R1” está baixa e se o tempo atual é posterior às 18:00. Dessa forma, se as condições são satisfeitas, a política evoca uma ação no roteador “R1” a fim de reservar 10% de sua banda passante ao perfil “QoS1”.

```
<use name="/policy">
 <add name="AdjustQoSPolicy">
  <use name="/template/policy">
   <create type="obligation"
   event= "/event/humanResourcesProcedureEvent"
   active="true">
     <arg name="R1_load"/>
     <arg name="daytime"/>
     <condition>
      <and>
       <eq>!R1_load;<!-- -->low</eq>
       <gt>!daytime;<!-- -->18:00</gt>
      </and>
     </condition>
     <action>
      <use name="/routers/R1">
       <modify profile=”QoS1” value=”10%”/>
      </use>
     </action>
   </create>
  </use>
 </add>
</use>
```

O módulo de manutenção de consistência recebe o dado "*Adjust QoS Policy*" (adj_qos_pol) e sua lista de justificativas, composta por "*Human Resources Procedure Event received*" (hr_proc_evt),"*R1 Low Load matched*" (R1_load_mat) e "*Daytime matched*" (dt_mat). A justificativa "*Human Resources Procedure Event received*" representa a recepção de evento do tipo *Human Resource Prodcedure*, a justificativa "*R1 Low Load matched*" representa que a carga do roteador R1 está baixa, e a justificativa "*Daytime matched*" representa que o período de tempo definido pela política foi verificado. O trecho de código abaixo apresenta uma representação interna desse dado e de suas justificativas.

```
justification(hr_proc_evt).
justification(R1_load_mat).
justification(dt_mat).
justificationIsPresent(X) :- generated(X).
justificationIsPresent(X) :- received(X).
datum(adj_qos_pol) :-
 justificationIsPresent(hr_proc_evt),
 justificationIsPresent(R1_load_mat),
 justificationIsPresent(dt_mat).
datumIsInternal(adj_qos_pol) :-
 generated(hr_proc_evt),
 generated(R1_load_mat),
 generated(dt_mat).
```

Uma situação pode ser apresentada como exemplo deste estudo de caso: devido à falta de sincronização dentro do grupo de *peers*, alguns *peers* entram em desacordo sobre a avaliação da condição temporal (tempo atual é posterior às 18:00). Essa situação pode tornar o grupo de *peers* inconsistente. A proposta descrita na presente dissertação define que módulos de serviço de gerenciamento (representado nesse estudo de caso pelo componente de processamento de políticas) informam mudanças de crença para o módulo de manutenção de consistência, o qual difunde essas mudanças dentro do grupo de *peers* conforme descrito no capítulo 3. Essa difusão poderia ser utilizada para evitar que o desacordo sobre uma questão temporal entre *peers* torne o grupo de *peers* inconsistente. O código abaixo demonstra a resposta produzida pelo módulo de manutenção de consistência nesta situação.

```
adj_qos_pol:external (hr_proc_evt:mod R1_load_mat:mod dt_mat:msg)
```

A resposta produzida pelo módulo de manutenção de consistência indica que o estado do dado de gerenciamento "*Adjust QoS Policy*" (adj_qos_pol) é "in" e marcado com a opção "external". Além disso, pode ser percebido que a presença da justificativa "*Daytime matched*" (dt_mat) foi verificada por meio do recebimento de mensagens de mudanças de crenças (indicada pelo ":msg" no trecho de código). Dessa forma, o estado do dado de gerenciamento se mantém consistente dentro do grupo de *peers* e o sistema ANM baseado em P2P apresenta um comportamento coerente.

A manutenção de consistência do estado das políticas em sistemas ANM descentralizados é tradicionalmente realizada por meio de entidades centralizados, tais como repositórios externos (MARQUEZAN et al., 2008). Em sistemas ANM baseados em P2P, essa centralização é normalmente realizada pela utilização de *super peers* (KAMIENSKI et al., 2008) (FALLON et al., 2007). A manutenção da consistência através de entidades centralizadoras apresenta dificuldade em relação a um bom desempenho em itens como escalabilidade e robustez e ainda impõe dificuldades na integração de diversas fontes de informação.

O presente trabalho também possibilita a utilização de um dado como justificativa para outro dado (conforme apresentado na Seção 3), funcionalidade chamada de *Justificativa*

Hierárquica. Neste exemplo, a justificativa "*R1 low load mat*" pode ser estendida para incluir diferentes "cargas" de diferentes recursos do roteador R1, tais como memória e processador. Assim, é definido o dado "*R1 Low Load matched*" (R1_load_mat) e sua lista de justificativas, composta por "*Cpu Low Load matched*" (cpu_load_mat) e "*Memory Low Load matched*" (mem_load_mat). O código abaixo demonstra a representação desse dado e suas justificativas.

```
justification(cpu_load_mat).
justification(mem_load_ma).
justificationIsPresent(X) :- generated(X).
justificationIsPresent(X) :- received(X).
datum(R1_load_mat) :-
 justificationIsPresent(cpu_load_mat),
 justificationIsPresent(mem_load_mat).
datumIsInternal(R1_load_mat) :-
 generated(cpu_load_mat),
 generated(mem_load_mat).
```

Após esta modificação, também é necessária uma pequena mudança na representação do dado "*Adjust QoS Policy*" (adj_qos_pol) e na regra que representa se o mesmo está "internal" (*datumIsInternal (adj_qos_pol)*). O dado "*Adjust QoS Policy*" agora é justificado pelo dado "*R1 Low Load matched*" (R1_load_mat). Além disso, a regra que representa se o "*Adjust QoS Policy*" está "internal" é justificada pela regra que define se o dado "*R1 Low Load matched*" está "internal" (*datumIsInternal(R1_load_mat)*). O código abaixo mostra a nova versão desse dado.

```
datum(adj_qos_pol) :-
 justificationIsPresent(hr_proc_evt),
 datum(R1_load_mat),
 justificationIsPresent(dt_mat).
datumIsInternal(adj_qos_pol) :-
 generated(hr_proc_evt),
 datumIsInternal(R1_load_mat),
 generated(dt_mat).
```

O suporte à utilização hierárquica de justificativas aprimora a representação de dados complexos conforme apresentado no último exemplo. Dessa forma, podem ser construídos dados de gerenciamento com maior granularidade sem, no entanto, prejuízos para a compreensão dos mesmos, devido à possibilidade de representação hierárquica.Essa construção também independe da fonte da presença das justificativas, o que auxilia na definição de dados de gerenciamento integrados.

Justificativas podem ser utilizadas para prover explicações para o usuário (KAGAL; HANSON; WEITZNER, 2008), e, usando justificativas hierárquicas, as mesmas podem aprimorar significativamente a compreensão de dados de gerenciamento (NOBRE; GRANVILLE, 2009a) (NOBRE; GRANVILLE, 2010). No contexto de PBNM, explicações para os estados de ativação de políticas permitem maior precisão na verificação sobre os resultados obtidos na aplicação de políticas, aumento na confiança do processo de decisão e aplicação de políticas e uma melhor avaliação sobre a correção de políticas (KAGAL; HANSON; WEITZNER, 2008).

Os estudos de caso apresentados demonstram as possibilidades e a aplicabilidade da proposta apresentada no capítulo 3. Além disso, a manutenção de consistência do estado dos dados de gerenciamento é realizada mantendo uma abordagem P2P. Também é possível verificar que esse resultado pode ser generalizado para outros ANM descentralizados, em que *peers* são representados por AMEs e grupos de *peers* são representados por AMDs (NOBRE; GRANVILLE, 2010).

4.4 Simulação

A avaliação das estratégias de comunicação descritas na presente dissertação (conforme o capítulo 3) pode ser realizada de diferentes formas. Em geral, avaliar propriedades de sistemas P2P é uma tarefa que apresenta diversos desafios. Esses sistemas podem variar significativamente em escala (*i.e.*, número de *peers*), e, ainda, a participação de *peers* no *overlay* é tipicamente dinâmica (*i.e.*, *peers* podem entrar e sair do *overlay*). Esses desafios e dificuldades estão presentes na avaliação de sistemas ANM baseado em P2P. Além disso, a avaliação de uma nova proposta para esses sistemas em um ambiente real possui diversas dificuldades.

A abordagem escolhida para avaliação das estratégias de comunicação descritas na presente dissertação foi a realização de experimentos por meio de *simulação*. Essa abordagem foi escolhida a fim de possibilitar um ambiente totalmente controlado para essa avaliação. Os experimentos simulados foram realizados para apresentar resultados que demonstrem as propriedades dessas estratégias em relação à escalabilidade e robustez.

Escalabilidade e robustez são algumas das métricas mais importantes que motivam a utilização de descentralização na infraestrutura de diferentes sistemas (MCCANN; HUEBSCHER, 2004) (BABAOGLU et al., 2006), tais como, por exemplo, sistemas ANM. Essas métricas podem ser verificadas, por exemplo, observando-se o número de mensagens trocadas para a manutenção da consistência do estado de um dado de gerenciamento em sistemas descentralizados (*e.g.*, sistemas ANM baseados em P2P). Também é importante avaliar a relação entre esse número de mensagens e outras variáveis, tais como o número de entidades que compartilham esse dado de gerenciamento (*peers* em sistemas ANM baseados em P2P).

Os experimentos simulados foram implementados utilizando o *PeerSim* (JELASITY et al., 2010), um simulador de sistemas P2P de código aberto, disponibilizado pela licença GPL (STALLMAN, 2007). O simulador é escrito na linguagem Java, sendo a mesma também utilizada para a implementação dos experimentos simulados. O PeerSim foi desenvolvido para possibilitar dinamicidade e escalabilidade nos experimentos simulados. Esse simulador também pode ser estendido com componentes e possui um mecanismo de configuração flexível. Em função destas características, existe uma quantidade bastante significativa de trabalhos científicos que utilizam esse simulador.

O PeerSim é composto por duas *engines* de simulação, uma simplificada, baseada em ciclos, e uma controlada por eventos. A *engine* baseada em ciclos utiliza algumas premissas para simplificar as simulações, tais como ignorar detalhes dos protocolos de comunicação da camada de transporte. A *engine* baseada em eventos é menos eficiente, mas mais realística, utilizando mais detalhes nas simulações. Protocolos baseados em ciclos também podem ser executados pela *engine* baseadas em eventos.

Os experimentos foram desenvolvidos por meio da utilização da *engine* baseada em eventos e utilizam um modelo de camada de transporte que permite emular algumas características encontradas em ambientes operacionais, tais como probabilidade de perda e atraso de mensagens. Nos experimentos realizados, o atraso de mensagens é uma variável controlada e o número de mensagens transmitidas é considerado um indicador de tráfego no *overlay* P2P. Apesar do experimento ser desenvolvido por meio da utilização da *engine* baseada em eventos, as medições apresentadas são realizadas em ciclos do simulador.

Os experimentos utilizam um protótipo de sistema ANM baseado em P2P que possui a habilidade de simular defeitos em *peers* e a construção do *overlay* é realizada aleatoriamente. Todos os *peers* dentro de cada grupo de *peers* executam a mesma estratégia de comunicação. Um *peer* é escolhido aleatoriamente como fonte primária de mudanças de

crenças a fim dessa fonte não influir nos resultados dos experimentos. Cada experimento foi conduzido, ao menos, por 10 vezes.

O restante da seção apresenta a avaliação das estratégias de comunicação descritas na presente dissertação (conforme o capítulo 3). A primeira estratégia de comunicação avaliada é a replicação livre. A segunda estratégia de comunicação avaliada é a replicação controlada.

4.4.1 Replicação Livre

A primeira estratégia de comunicação avaliada é a replicação livre. Essa estratégia de comunicação é avaliada por meio de dois experimentos. Nesses experimentos, o número de *peers* do grupo de *peers* é utilizado como variável independente. Os números de *peers* escolhidos para os experimentos foram definidos intuitivamente utilizando os estudos de caso descritos na seção 4.3 como base. Pode ser observado que não são esperados grupos de *peers* com grande número de *peers* participantes em sistemas ANM baseados em P2P. A variância observada nesses experimentos foi baixa.

No primeiro experimento, é medido o número de mensagem trocadas para replicar mudanças de justificativas dentro de um grupo de *peers*. Esse número deve ser considerado como um importante custo da operação do grupo de *peers*, assim, é importante para análises de escalabilidade. Nesse experimento não existe perda de mensagens dentro do grupo de *peers* (*i.e.*, probabilidade de perda de mensagens é 0%). Os resultados são apresentados na Figura 4.3.

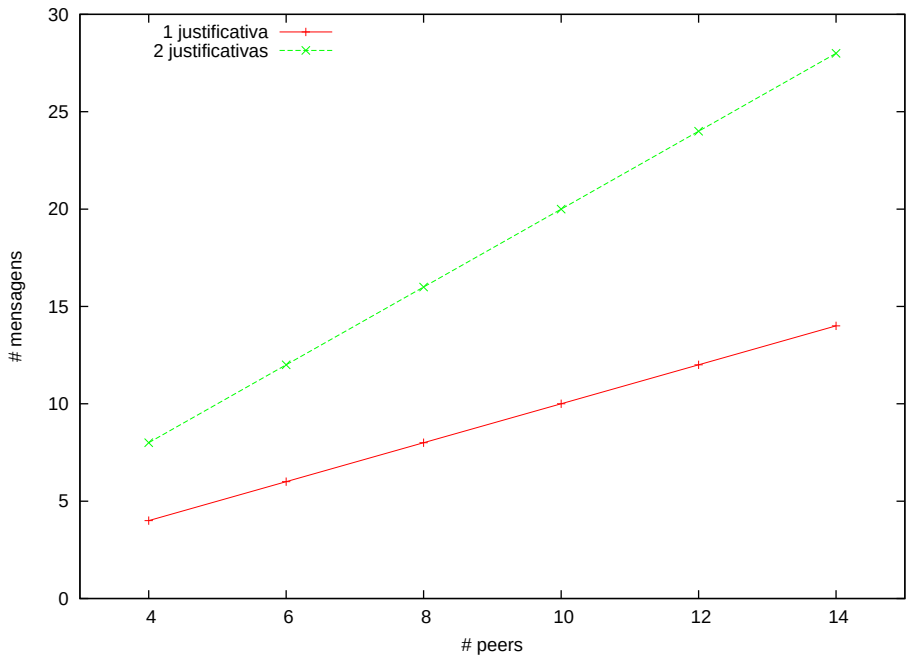

Figura 4.3: Troca de mensagens devido a mudanças em justificativas

A Figura 4.3 apresenta resultados para o número de mensagens trocadas para replicar mudanças de justificativas em função do número de *peers* do grupo de *peers*. São apresentados resultados para a troca de mensagens relativa a dois valores: 1 e 2 justificativas (valores representados pela diferentes curvas nessa figura). Os valores variam de 4 a 14 mensagens para 1 justificativa e de 8 a 28 mensagens para 2 justificativas. O

número de mensagens trocadas cresce linearmente com o número de *peers* participantes. Assim, pode ser inferida a tendência de comportamento de grupos de *peers* com um maior número de *peers* participantes.

A replicação livre mostra propriedades aceitáveis em relação ao número de mensagens trocadas, já que a operação está restrita a cada grupo de *peers*, e o tamanho desses grupos é condizente com os estudos de caso apresentados na seção 4.3. Dessa forma, grupos de *peers* também funcionam como abstrações de agregação que possibilitam a escalabilidade na troca de mensagens. Diversos sistemas utilizam abstrações semelhantes para restringir a troca de mensagens, como, por exemplo, "*Domains*" no sistema *Astrolabe* (VAN RENESSE; BIRMAN; VOGELS, 2003).

Evidentemente, a replicação livre não é uma a estratégia de comunicação eficiente para a operação de grupo de *peers* com grande número de *peers* participantes, apesar da mesma possuir características desejáveis em relação à tolerância a falhas na troca de mensagens. Assim, modificações nessa estratégia são necessárias para uma operação eficiente em relação ao número de mensagens trocadas para a difusão de justificativas, tais como a utilização de protocolos baseados em *gossip* (GUPTA; BIRMAN; RENESSE, 2002).

O segundo experimento foi realizado para determinar a influência da perda de mensagens na disseminação de uma mudança de justificativa. Nesse experimento, a variável relativa à probabilidade de perda de mensagens é modificada com os seguintes valores: 25%, 50% e 75% (respectivamente, 0,25, 0,5, e 0,75, como indicado na Figura 4.4). A Figura 4.4 apresenta o percentual médio de *peers* coerentes (e corretos) depois da troca de mensagens parar de ocorrer.

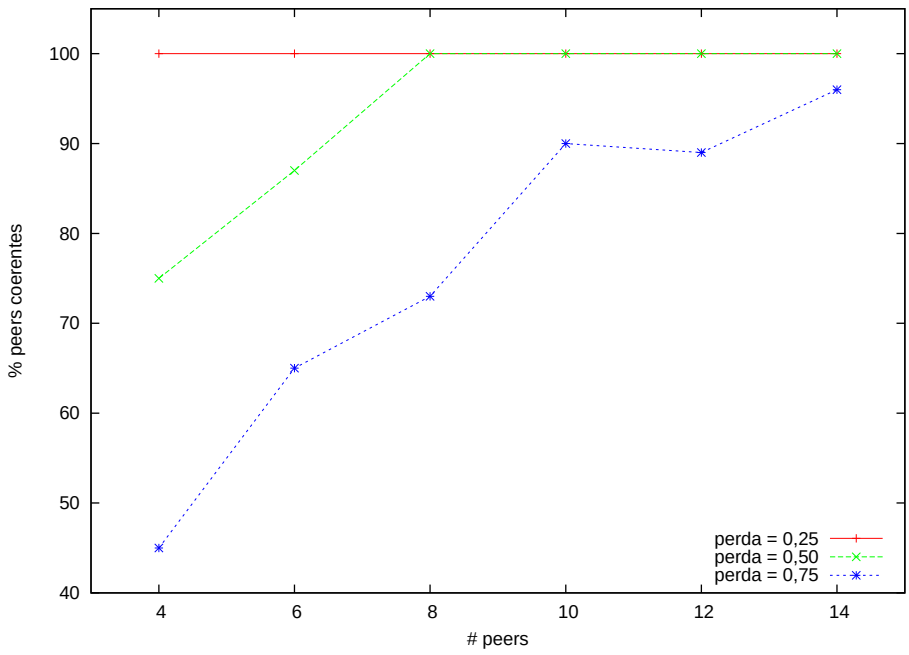

Figura 4.4: *Peers* coerentes após uma mudança de justificativa

A Figura 4.4 apresenta resultados para o percentual de *peers* coerentes após a replicação de uma mudança de justificativa em função do número de *peers* do grupo de *peers*. São apresentados resultados relativos a três valores para probabilidade de perda de mensagens: 0,25, 0,5, e 0,75 (valores representados pela diferentes curvas nessa figura). Os

valores variam de, aproximadamente, 45% a 96% de *peers* coerentes com uma probabilidade de perda de mensagens de 75%, e de, aproximadamente, 75% a 100% de *peers* coerentes com uma probabilidade de perda de mensagens de 50%. Os valores se mantêm em 100% de *peers* coerentes com uma probabilidade de perda de mensagens de 25%.

O experimento mostra a influência da perda de mensagens na replicação livre. Como pode ser visto por meio dos resultados demonstrados na Figura 4.4, o percentual de *peers* coerentes cresce com o número de *peers* participantes (ou se mantém em 100%), e altas probabilidades de perdas levam a uma menor consistência no grupo de *peers*. No entanto, mesmo com um pequeno número de *peers* participantes, o percentual de *peers* coerentes é substancial. Além disso, um maior número de *peers* participantes no grupo de *peers* diminui a influência da probabilidade de perda de mensagens.

Um dos estudos de casos apresentados na seção 4.3 (o gerenciamento de falhas em enlaces de rede Ethernet em provedores de serviços) pode representar um cenário em que são observadas altas taxas de perda de mensagem. Dessa forma, as probabilidades simuladas (especialmente 75%) ocorrem com frequência devido a equipamentos ou enlaces de rede defeituosos ou sobrecarregados. Esse estudo de caso é focado em Gerenciamento de Falhas, considerando o modelo FCAPS (*Fault, Configuration, Accounting, Performance, Security*), assim, o sistema ANM baseado em P2P precisa ter um desempenho aceitável em condições de rede desfavoráveis. Ainda é importante mencionar que este cenário considera uma infraestrutura IP sobreposta.

A operação do grupo de *peers* não é significativamente sensível à perda de mensagens; dessa forma, os resultados mostram uma maior tolerância a falhas na troca de mensagens. Pode-se verificar também que um aumento no número de *peers* dentro do grupo de *peers* diminui a sensibilidade do mesmo frente à perda de mensagens. Esses resultados estão relacionados com as funcionalidades introduzidas pelo processo de replicação.

Um aumento no número de *peers* leva, no entanto, a um aumento no número de mensagens trocadas para a difusão de uma nova crença, então as vantagens relativas à robustez são conseguidas com algum custo. De qualquer forma, mensagens são trocadas apenas dentro do grupo de *peers*, e um alto número de *peers* em um grupo de *peers* não é esperado na maior parte das tarefas de gerenciamento. Assim, o número de mensagens trocadas não impõe problemas em relação à escalabilidade.

4.4.2 Replicação Controlada

Experimentos foram realizados a fim de avaliar a replicação controlada como estratégia de comunicação para a troca de crenças. Já que a replicação controlada é utilizada, é necessário descrever os parâmetros da Função de Controle da Replicação (*Replication Controlling Function* - RCF) em cada experimento. Além disso, todos os *peers* do grupo de *peers* utilizam exatamente os mesmos parâmetros para a RCF. Foram simulados até 100.000 *peers* em um grupo de *peers*.

No primeiro experimento (ρ = 1, T = 3) é medido o número de mensagens trocadas para replicar uma mudança de justificativa dentro de um grupo de *peers*. Esse número deve ser considerado como um importante custo da operação do grupo de *peers*, assim, é importante para análises de escalabilidade. Nesse experimento, não existem falhas em *peers* ou na troca de mensagens. Os resultados desse experimento são apresentados na Figura 4.5, onde os mesmos são descritos por meio dos valores médios medidos e com intervalo de confiança de 95% para cada tamanho de grupo de *peers*.

A Figura 4.5 apresenta resultados para o número de mensagens trocadas para replicar uma mudança de justificativa em função do número de *peers* do grupo de *peers*, utili-

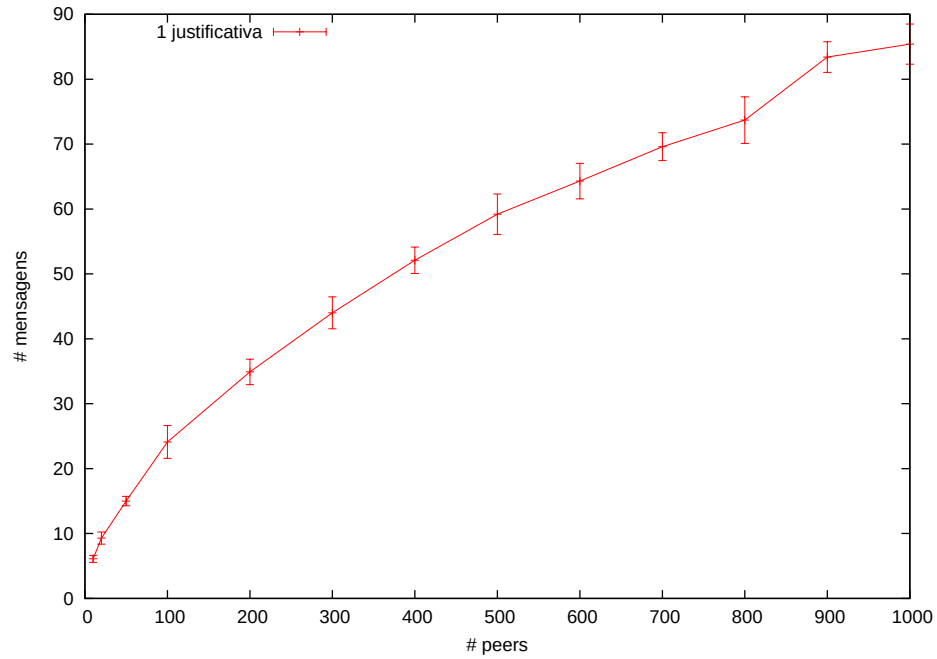

Figura 4.5: Troca de mensagens devido a mudanças em justificativas. As barras de erro indicam as médias e o intervalo de confiança de 95%

zando replicação controlada. Os valores representados pela curva nessa figura variam de, aproximadamente, 6 mensagens em grupos com 10 *peers* à 85 mensagens em grupos com 1000 *peers*.

A estratégia de comunicação demonstra características de escalabilidade aceitáveis no número de mensagens trocadas dentro do grupo de *peers*. Os resultados do experimento apresentados na Figura 4.5 demonstram que essa estratégia não apresenta problemas de estabilidade ou convergência. A razão principal para issto é que a RCF escolhida (ρ = 1, T = 3) é eficiente para a operação em grandes grupo de *peers*.

Grupos de *peers* com número elevado de participantes não são esperados na maioria das tarefas realizadas em um sistema ANM baseado em P2P (e em outros sistemas descentralizados). No entanto, é importante avaliar esta situação para inferir a tendência de comportamento em grandes grupos de *peers* e, assim, verificar a viabilidade dos mesmos.

O segundo experimento foi realizado para determinar a influência da perda de mensagens na disseminação de uma mudança de justificativa. Nesse experimento, foi variada a probabilidade de perda de mensagens com os seguintes valores: 50% e 75%. Durante os experimentos, foi verificada uma correlação significativa entre a perda de mensagens e a constante positiva de proliferação (ρ) nos resultados. Assim, ρ foi também variado nesse experimento com os seguintes valores: 0,5 e 0,25. Na Figura 4.6 é apresentado o percentual médio de *peers* coerentes (e corretos).

A Figura 4.6 apresenta resultados para o percentual de *peers* coerentes após a replicação de uma mudança de justificativa em função do número de *peers* do grupo de *peers*, e esse número varia entre 5 e 50 *peers*. Os valores representados para ρ igual a 0,5 nessa figura variam de, aproximadamente, 73% a 100% de *peers* coerentes com probabilidade de perda de mensagens de 50%, e de, aproximadamente, 49% a 99% de *peers* coerentes com probabilidade de perda de mensagens de 75%. Os valores para ρ igual a 0,25 variam de, aproximadamente, 73% a 100% de *peers* coerentes com probabilidade de perda de mensa-

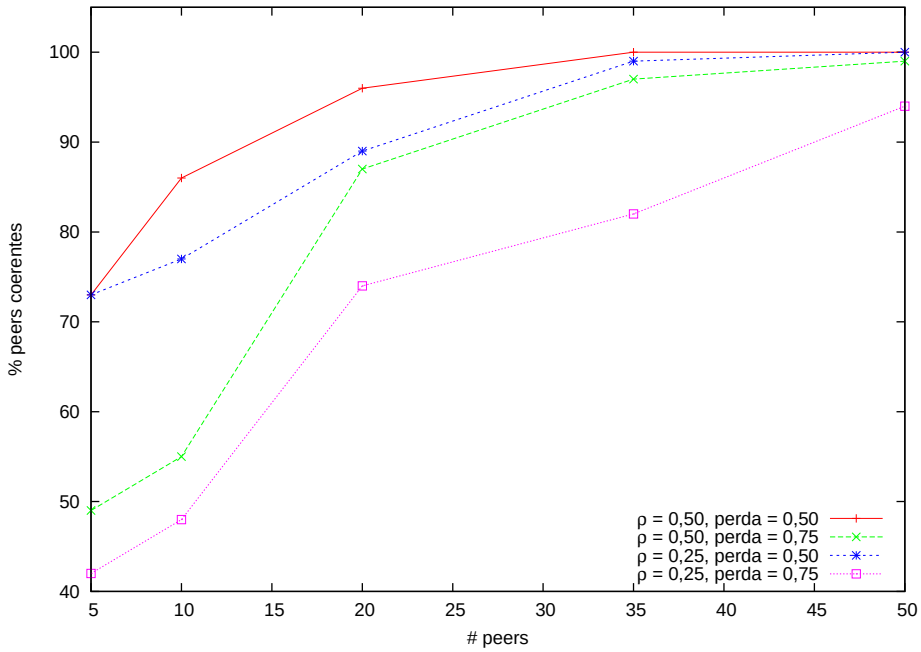

Figura 4.6: *Peers* coerentes após uma mudança de justificativa

gens de 50%, e de, aproximadamente, 42% a 94% de *peers* coerentes com probabilidade de perda de mensagens de 75%.

O experimento mostra a influência da perda de mensagens e da constante positiva de proliferação (ρ) no estratégia de comunicação utilizada nesse experimento, a replicação controlada. Como pode ser visto pelos resultados na Figura 4.6, altas probabilidades de perda levam a uma menor consistência no grupo de *peers*, mas, mesmo com um número reduzido de *peers* participantes, o percentual de *peers* coerentes é substancial. Aumentos no número de *peers* participantes no grupo de *peers* e nos valores de ρ também diminuem a influência da probabilidade de perda de mensagem.

5 CONCLUSÃO E TRABALHOS FUTUROS

A complexidade crescente das redes de computadores demanda soluções sofisticadas para gerenciar a infraestrutura de comunicação subjacente e auxiliar os administradores de rede em suas tarefas diárias. O suporte dessas demandas pelos diversos modelos de gerenciamento de redes é um tópico de pesquisa fundamental na área de gerenciamento de redes. A fim de prover um suporte eficiente para essas tarefas, é necessário que o sistema de gerenciamento de redes exiba algumas funcionalidades de descentralização e automação.

Funcionalidades de descentralização e automação podem ser proporcionadas por sistemas de Gerenciamento Autonômico de Redes (*Autonomic Network Management* - ANM) descentralizados. Uma possibilidade interessante de infraestrutura para sistema ANM é a utilização de *overlays* P2P. No entanto, a consistência dos estados dos dados de gerenciamento representa um desafio para esses sistemas. Em geral, essa consistência é mantida por meio da introdução de entidades centralizadoras (*e.g.*, repositórios externos), o que desperdiça características desejáveis da descentralização, tais como escalabilidade e robustez.

Esta dissertação empregou funcionalidades de *Manutenção da Verdade Multiagente* para aprimorar a manutenção da consistência do estado dos dados de gerenciamento (*e.g.*, estados de políticas) em sistemas ANM baseados em P2P. Essa manutenção considera que os *peers* são organizados em grupo de *peers* de acordo com os dados que os *peers* compartilham. O trabalho também demonstrou que este resultado pode ser estendido para outros sistemas ANM descentralizados. Essa extensão considera que *peers* possuem características análogas com Elementos de Gerenciamento Autonômico (*Autonomic Management Elements* -AMEs) e grupos de *peers* possuem características análogas a Domínios de Gerenciamento Autonômico (*Autonomic Management Domains* - AMDs).

O conceito principal empregado para a manutenção de consistência é a utilização de *justificativas*. Dessa forma, um dado de gerenciamento é acreditado se o mesmo possui justificativas válidas. (*i.e.*, razões válidas), ou seja, a crença sobre o estado de um dado é fundamentada nas suas justificativas. As justificativas são produzidas em uma forma definida e, de acordo com as informações disponíveis, geradas internamente ao *peer* ou recebidas por meio do grupo de *peers*. Na implementação apresentada, os dados e suas justificativas são definidos por fatos e regras na linguagem ISO Prolog.

Estratégias de comunicação para a troca de crenças dentro de um grupo de *peers* são também propostas na presente dissertação. Essa troca é modelada utilizando-se conceitos oriundos de modelos de computação distribuída inspirados na Biologia, especialmente modelos baseados em proliferação devido aos requisitos de comunicação da presente proposta. Dentre os modelos baseados em proliferação, o presente trabalho se apoia em características encontradas na *replicação*. Na proposta apresentada nessa dissertação, a

replicação é representada por meio da difusão de mensagens para replicar mudanças em crenças entre os *peers* de um grupo de *peers*, as quais podem ser controladas por parâmetros da própria difusão.

A implementação da utilização de justificativas e da troca de crenças dentro do grupo de *peers* é realizada por meio de um *módulo de manutenção de consistência* que roda em cada um dos *peers*. Esse módulo trabalha associando dados de gerenciamento e suas respectivas justificativas, utilizando os serviços de comunicação do *overlay* P2P para difundir as mudanças de crenças. Os serviços desse módulo são descritos através de uma interface simples, a qual é descrita na presente dissertação.

Estudos de casos são descritos para demonstrar as possibilidades e a aplicabilidade do presente trabalho. Os estudos escolhidos foram o suporte para o gerenciamento colaborativo de falhas em enlaces de acesso e a utilização descentralizada de políticas de obrigação. Nesses estudos, foi possível concluir que a manutenção de consistência dos estados dos dados de gerenciamento por meio de funcionalidades de Manutenção da Verdade Multiagente é verossímil e interessante. Como essas funcionalidades são realizadas mantendo a abordagem P2P, características desejáveis dessa abordagem também são preservadas. Os estudos de caso também demonstram que a proposta pode ser generalizada de sistemas ANM baseados em P2P para outros sistemas ANM descentralizados.

O trabalho também apresenta avaliações por meio da realização de experimentos com simulação. Nos experimentos simulados, são avaliados aspectos relacionados com a escalabilidade e robustez das estratégias de comunicação frente a características do *overlay* P2P, tais como o tamanho do grupo de *peers*, e também características da infraestrutura de comunicações, tais como a probabilidade de perda de mensagens. Nesses experimentos foi possível concluir que as estratégias de comunicação possuem bom desempenho nesses aspectos. Modificações nos parâmetros relacionados às estratégias de comunicação foram também avaliadas. Pode-se concluir que essas modificações têm influência significativa no desempenho das estratégias de comunicação.

Como trabalhos futuros, otimizações nas estratégias de comunicação podem ser implementados a fim de aumentar a eficiência da difusão de mensagens no grupo de *peers*. Por exemplo, esquemas de compressão podem ser empregados para agregar diversas mudanças de crença em apenas uma mensagem a ser difundida. Essas otimizações poderiam ser realizadas por meio da utilização de características encontradas no processo de replicação, como propriedades do *overlay* P2P e particularidades dos dados compartilhados no grupo de *peers*.

Aspectos de segurança também poderiam ser investigados na difusão de mudanças de crença dentro do grupo de *peers*. Uma das questões que poderia ser investigada é a utilização de mensagens assinadas para evitar que *peers* maliciosos introduzam mudanças de crença perniciosas. É importante, no entanto, que novas funcionalidades relacionadas a aspectos de segurança mantenham as características de descentralização desejáveis encontradas em sistemas ANM baseados em P2P.

REFERÊNCIAS

ANDROUTSELLIS-THEOTOKIS, S.; SPINELLIS, D. A Survey of Peer-to-Peer Content Distribution Technologies. **ACM Computing Surveys**, New York, NY, USA, v.36, n.4, p.335–371, 2004.

BABAOGLU, O. et al. Design patterns from biology for distributed computing. **ACM Transactions on Autonomous and Adaptive Systems (TAAS)**, New York, NY, USA, v.1, n.1, p.26–66, September 2006.

BINZENHOFER, A. et al. A P2P-Based Framework for Distributed Network Management. In: WIRELESS SYSTEMS AND NETWORK ARCHITECTURES IN NEXT GENERATION INTERNET, Heidelberg, Germany. **Proceedings...** Springer Berlin, 2006. p.198–210. (Lecture Notes in Computer Science, v.3883).

CHAPARADZA, R. UniFAFF: a unified framework for implementing autonomic fault management and failure detection for self-managing networks. **International Journal of Network Management**, New York, NY, USA, v.19, n.4, p.271–290, August 2009.

CLARK, K. L.; MCCABE, F. G. Ontology schema for an agent belief store. **International Journal of Human-Computer Studies**, Duluth, MN, USA, v.65, n.7, p.640–658, July 2007.

DAMIANOU, N. et al. **Ponder2 - The Ponder2 Policy Environment**. Disponível em: <`http://www.ponder2.net/`>. Acesso em Agosto de 2010.

DIMOPOULOS, Y.; MORAITIS, P. Multi-Agent Coordination and Cooperation through Classical Planning. In: IEEE/WIC/ACM INTERNATIONAL CONFERENCE ON INTELLIGENT AGENT TECHNOLOGY (IAT), Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2006. p.398–402.

DOYLE, J. A truth maintenance system. **Computation & intelligence: collected readings**, Menlo Park, CA, USA, p.529–554, November 1979.

EMANICS. **Activities - Work Package 9 - Autonomic Management**. Disponível em: <`http://emanics.org/content/view/63/109/`>. Acesso em Agosto de 2010.

FALLON, L. et al. Self-forming Network Management Topologies in the Madeira Management System. **Lecture Notes in Computer Science**, Heidelberg, Germany, v.4543, p.61, 2007.

FIORESE, A.; SIMÕES, P.; BOAVIDA, F. A P2P-Based Approach to Cross-Domain Network and Service Management. In: INTERNATIONAL CONFERENCE ON AUTONOMOUS INFRASTRUCTURE, MANAGEMENT AND SECURITY (AIMS), 3., Heidelberg, Germany. **Proceedings...** Springer Berlin, 2009. p.182.

FISCHER, M.; LYNCH, N.; MERRITT, M. Easy impossibility proofs for distributed consensus problems. In: SIMONS, B.; SPECTOR, A. (Ed.). **Fault-Tolerant Distributed Computing**. Heidelberg, Germany: Springer Berlin, 1990. p.147–170. (Lecture Notes in Computer Science, v.448).

FISCHER, M.; LYNCH, N.; PATERSON, M. Impossibility of distributed consensus with one faulty process. **Journal of the ACM (JACM)**, New York, NY, USA, v.32, n.2, p.374–382, 1985.

GOLDSZMIDT, G.; YEMINI, Y. Distributed management by delegation. In: INTERNATIONAL CONFERENCE ON DISTRIBUTED COMPUTING SYSTEMS, 15., Washington, DC, USA. **Proceedings...** IEEE Computer Society, 1995. p.333–340.

GONG, L. JXTA: a network programming environment. **IEEE Internet Computing**, Washington, DC, USA, v.5, n.3, p.88–95, June 2001.

GRANVILLE, L. Z. et al. Managing computer networks using peer-to-peer technologies. **IEEE Communications Magazine**, Washington, DC, USA, v.43, n.10, p.62–68, 2005.

GUPTA, I.; BIRMAN, K.; RENESSE, R. van. Fighting fire with fire: using randomized gossip to combat stochastic scalability limits. **Quality and Reliability Engineering International**, New York, NY, USA, v.18, n.3, p.165–184, 2002.

HARRINGTON, D.; PRESUHN, R.; WIJNEN, B. **An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks**. Fremont, CA, USA: IETF, 2002. n.3411. (Request for Comments).

HORLING, B.; LESSER, V. A survey of multi-agent organizational paradigms. **The Knowledge Engineering Review**, New York, NY, USA, v.19, n.04, p.281–316, 2005.

HU, J. et al. Multi-agent system based autonomic computing environment. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING AND CYBERNETICS, Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2004. v.1, p.105–110.

HUHNS, M. N.; BRIDGELAND, D. M. Multiagent truth maintenance. **IEEE Transactions on Systems, Man and Cybernetics**, Washington, DC, USA, v.21, n.6, p.1437–1445, 1991.

HUNT, P. **ZooKeeper**: a distributed coordination service for distributed applications. Disponível em: <`http://wiki.apache.org/hadoop/ZooKeeper`>. Acesso em Agosto de 2010.

JELASITY, M. et al. **The Peersim Simulator**. Disponível em: <`http://peersim.sf.net`>. Acesso em Agosto de 2010.

JENNINGS, B. et al. Towards autonomic management of communications networks. **IEEE Communications Magazine**, Washington, DC, USA, v.45, n.10, p.112–121, 2007.

Joint Technical Committee ISO/IEC. **ISO/IEC 9596, Information Technology, Open Systems Interconnection, Common Management Information Protocol (CMIP) – Part 1**: specification. Geneva, Switzerland: International Organisation for Standardisation, 1991.

KAGAL, L.; HANSON, C.; WEITZNER, D. Using Dependency Tracking to Provide Explanations for Policy Management. In: IEEE WORKSHOP ON POLICIES FOR DISTRIBUTED SYSTEMS AND NETWORKS (POLICY), Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2008. p.54–61.

KAMIENSKI, C. et al. PBMAN: a policy-based management framework for ambient networks. In: IEEE WORKSHOP ON POLICIES FOR DISTRIBUTED SYSTEMS AND NETWORKS (POLICY), Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2006. p.79–83.

KAMIENSKI, C. et al. Design and implementation of a policy-based management framework for Ambient Networks: choices and lessons learned. In: IEEE/IFIP NETWORK OPERATIONS AND MANAGEMENT SYMPOSIUM (NOMS), Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2008. p.775–778.

KARIMI, O. et al. Availability in Peer to Peer Management Networks. In: ASIA-PACIFIC SYMPOSIUM ON NETWORK OPERATIONS AND MANAGEMENT (APNOMS), 11., Heidelberg, Germany. **Proceedings...** Springer-Verlag Berlin, 2008. p.555.

KEPHART, J. O.; CHESS, D. M. The Vision of Autonomic Computing. **Computer**, Los Alamitos, CA, USA, v.36, n.1, p.41–50, 2003.

KIND, A. et al. Advanced network monitoring brings life to the awareness plane. **IEEE Communications Magazine**, Washington, DC, USA, v.46, n.10, p.140–146, 2008.

KONSTANTINOU, A.; YEMINI, Y. A2A: an architecture for autonomic management coordination. In: IFIP/IEEE INTERNATIONAL WORKSHOP ON DISTRIBUTED SYSTEMS: OPERATIONS AND MANAGEMENT (DSOM), Heidelberg, Germany. **Proceedings...** Springer-Verlag Berlin, 2009. p.85.

KOUBARAKIS, M. Multi-agent Systems and Peer-to-Peer Computing: methods, systems, and challenges. In: **Cooperative Information Agents VII**. Heidelberg, Germany: Springer Berlin, 2003. p.46–61. (Lecture Notes in Computer Science, v.2782).

LEINWAND, A.; CONDROY, K. F. **Network Management**: a practical perspective. 2.ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1996.

LIEBAU, N. et al. Token-based accounting for p2p-systems. In: KOMMUNIKATION IN VERTEILTEN SYSTEMEN (KIVS), Heidelberg, Germany. **Proceedings...** Springer Berlin, 2005. p.16–28.

LORENZI, F. A multiagent knowledge-based recommender approach with truth maintenance. In: ACM CONFERENCE ON RECOMMENDER SYSTEMS, New York, NY, USA. **Proceedings...** ACM Press, 2007. p.195–198.

LUPU, E. et al. AMUSE: autonomic management of ubiquitous systems for e-health. **J. Concurrency and Computation: Practice and Experience**, New York, NY, USA, 2007.

MARQUEZAN, C. C. et al. Self-managed services over a P2P-based Network Management Overlay. In: LATIN AMERICAN AUTONOMIC COMPUTING SYMPOSIUM (LAACS 2007), Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2007. p.7.

MARQUEZAN, C. et al. Maintenance of Monitoring Systems Throughout Self-healing Mechanisms. In: IFIP/IEEE INTERNATIONAL WORKSHOP ON DISTRIBUTED SYSTEMS: OPERATIONS AND MANAGEMENT (DSOM), 19., Heidelberg, Germany. **Proceedings...** Springer Berlin, 2008. p.188.

MARTIN-FLATIN, J.-P.; ZNATY, S.; HABAUX, J.-P. A Survey of Distributed Entreprise Network and Systems Management Paradigms. **Journal of Network and Systems Management**, New York, NY, USA, v.7, n.1, p.9–26, 1999.

MCCANN, J. A.; HUEBSCHER, M. C. Evaluation Issues in Autonomic Computing. In: GRID AND COOPERATIVE COMPUTING (GCC), Heidelberg, Germany. **Proceedings...** Springer Berlin, 2004. p.597–608. (Lecture Notes in Computer Science, v.3252).

MCFARLAND, M.; SALAM, S.; CHECKER, R. Ethernet OAM: key enabler for carrier class metro ethernet services. **IEEE Communications Magazine**, Washington, DC, USA, v.43, n.11, p.152–157, 2005.

MELCHIORS, C.; GRANVILLE, L. Z.; TAROUCO, L. M. R. P2P-Based Management of Collaboration Communication Infrastructures. In: SAMULI NIIRANEN JARI YLI-HIETANEN, A. L. (Ed.). **Open Information Management**: applications of interconnectivity and collaboration. Hershey, PA, USA: Information Science Reference, 2009.

MOORE, E. B. **Policy Core Information Model (PCIM) Extensions**. Fremont, CA, USA: IETF, 2003. n.3460. (Request for Comments).

MOORE, R. **Creating a DCOP Interface**. Disponível em: <`http://developer.kde.org/documentation/tutorials/dot/dcopiface/dcop-interface.html`>. Acesso em Agosto de 2010.

MORO, G.; OUKSEL, A. M.; SARTORI, C. Agents and Peer-to-Peer Computing: a promising combination of paradigms. In: AGENTS AND PEER-TO-PEER COMPUTING (AP2PC), Heidelberg, Germany. **Proceedings...** Springer Berlin, 2003. p.15–28.

NOBRE, J. C.; GRANVILLE, L. Z. Towards Consistency of Policy States in Decentralized Autonomic Network Management. In: IEEE INTERNATIONAL WORKSHOP ON POLICIES FOR DISTRIBUTED SYSTEMS AND NETWORKS (POLICY), Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2009. p.170–173.

NOBRE, J. C.; GRANVILLE, L. Z. Consistency of Policy States in Decentralized Autonomic Network Management. In: IEEE/IFIP NETWORK OPERATIONS AND MANAGEMENT SYMPOSIUM (NOMS), 12., Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2010.

NOBRE, J.; GRANVILLE, L. Consistency of States of Management Data in P2P-Based Autonomic Network Management. In: IFIP/IEEE INTERNATIONAL WORKSHOP ON DISTRIBUTED SYSTEMS: OPERATIONS AND MANAGEMENT (DSOM), 20., Heidelberg, Germany. **Proceedings...** Springer-Verlag Berlin, 2009. p.99.

OOI, B. C. et al. Information Acquisition Through an Integrated Paradigm: agent + peer-to-peer. In: AGENTS AND PEER-TO-PEER COMPUTING (AP2PC), Heidelberg, Germany. **Proceedings...** Springer Berlin, 2003. p.1–12.

PANISSON, A. et al. Designing the Architecture of P2P-Based Network Management Systems. In: IEEE SYMPOSIUM ON COMPUTERS AND COMMUNICATIONS (ISCC), 11., Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2006. p.69–75.

PENNINGTON, H. **D-bus specification**. Disponível em: <`http://dbus.freedesktop.org/doc/dbus-specification.html`>. Acesso em Agosto de 2010.

PRAS, A. et al. Key Research Challenges in Network Management. **IEEE communications magazine**, New York, NY, USA, v.45, n.10, p.104–110, October 2007.

RYOO, J. et al. OAM and its performance monitoring mechanisms for carrier ethernet transport networks. **IEEE Communications Magazine**, Washington, DC, USA, v.46, n.3, p.97–103, 2008.

SANCHEZ, R.; RAPTIS, L.; VAXEVANAKIS, K. Ethernet as a carrier grade technology: developments and innovations. **IEEE Communications Magazine**, Washington, DC, USA, v.46, n.9, p.88–94, September 2008.

SCHONWALDER, J.; PRAS, A.; MARTIN-FLATIN, J. P. On the future of Internet management technologies. **IEEE Communications Magazine**, Washington, DC, USA, v.41, n.10, p.90–97, 2003.

SCHöNWALDER, J.; QUITTEK, J.; KAPPLER, C. Building Distributed Management Applications with the IETF ScriptMIB. **IEEE Journal on Selected Areas in Communications**, Washington, DC, USA, v.18, n.5, p.702–714, 2000.

SCOWEN, R. **Technical Report ISO/IEC DIS 13211-1**: 1995 (e). Geneva, Switzerland: International Organisation for Standardisation, 1995.

SLOMAN, M. Policy driven management for distributed systems. **Journal of Network and Systems Management**, New York, NY, USA, v.2, n.4, p.333–360, 1994.

STALLMAN, R. M. **GNU General Public License Version 3**. Boston, MA, USA: Free Software Foundation, Inc., 2007.

TESAURO, G. et al. A multi-agent systems approach to autonomic computing. In: INTERNATIONAL JOINT CONFERENCE ON AUTONOMOUS AGENTS AND MULTI-AGENT SYSTEMS (AAMAS), 3., Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2004. p.464–471.

VAN RENESSE, R.; BIRMAN, K.; VOGELS, W. Astrolabe: a robust and scalable technology for distributed system monitoring, management, and data mining. **ACM Transactions on Computer Systems (TOCS)**, New York, NY, USA, v.21, n.2, p.164–206, 2003.

YALAGANDULA, P.; DAHLIN, M. A scalable distributed information management system. **ACM SIGCOMM Computer Communication Review**, New York, NY, USA, v.34, n.4, p.379–390, 2004.

APÊNDICE A ARTIGO PUBLICADO – POLICY 2009

Neste apêndice é apresentado o artigo “Towards Consistency of Policy States in Decentralized Autonomic Network Management”, desenvolvido durante a execução do trabalho descrito na presente dissertação. O artigo apresenta a proposta inicial de introdução de funcionalidades de manutenção da verdade multiagente em sistemas de Gerenciamento Autonômico de Redes descentralizados. Além disso, o artigo apresenta um esboço da utilização de replicação como estratégia de comunicação para prover suporte a essas funcionalidades. Experimentos simulados são realizados para avaliar a estratégia de comunicação apresentada. Finalmente, um estudo de caso relativo à manutenção da consistência do estado de ativação de políticas de gerenciamento de rede é descrito para mostrar a viabilidade dessa proposta.

- **Título**: *Towards Consistency of Policy States in Decentralized Autonomic Network Management*
- **Evento**: *10th IEEE International Symposium on Policies for Distributed Systems and Networks (POLICY 2009)*
- **URL**: http://www.policy-workshop.org/2009/
- **Data**: De 20 a 22 de julho de 2009
- **Local**: Londres, Inglaterra

Towards Consistency of Policy States in Decentralized Autonomic Network Management

Jéferson Campos Nobre, Lisandro Zambenedetti Granville
Federal University of Rio Grande do Sul – Institute of Informatics
Av. Bento Gonçalves, 9500 - Porto Alegre, RS - Brazil
{jcnobre,granville}@inf.ufrgs.br

Abstract

Autonomic network management is a vision that brings autonomic computing principles to network management. In this vision, the use of policies is a key aspect. Besides, it is necessary some level of decentralization to enable broad autonomic capabilities. However, the consistency of policy states among autonomic management elements is an important challenge. In this paper we introduce multi-agent truth maintenance features in decentralized autonomic network management as a mechanism to bring consistency maintenance of policy states. We developed a model of a decentralized autonomic network management system on Peersim to perform simulation experiments. Besides, the utilization of policies in P2P-based autonomic network management systems is presented as case study.

1 Introduction

The increasing complexity of computer networks requires sophisticated solutions to manage the underlying communication infrastructure and help network human administrators [7] in their daily tasks. The application of autonomic computing concepts in network management, known as *Autonomic Network Management* (ANM), has been proposed as a way to address some demands faced by traditional network management.

ANM systems can be implemented through different models of management distribution, from totally centralized management to highly decentralized ones. Some authors claim that some level of decentralization is required to achieve a more adequate ANM [7]. In this scenario, the typical design of decentralized ANM systems is based on a set of *Autonomic Management Elements* (AME) that execute management tasks and interact to form an Autonomic Management Domain (AMD) [7].

Policies play a key role in ANM systems [2] [7]. In a policy-enabled decentralized ANM system, AMEs must be policy-aware and take part in management decisions. These decisions are processed through the analysis of requests against the network policy.

A managed element can be controlled by one or multiple AMEs. In the latter case, policy states become distributed information. Inconsistency in this information could lead to improper operation of the ANM system. This failure could be caused by faults in network and computational resources or even due to lack of synchronization among AMEs.

Mechanisms to support consistency of policy states in current decentralized ANM systems are, in general, still supported by some centralized systems [3], missing opportunities of decentralized facilities. In other approaches, the mechanism to distribute policy states is not clearly stated [7].

In this paper, we propose a consistency maintenance module in each AME to improve the consistency of policy states in decentralized ANM systems. The module supports the improvement through features inspired by multi-agent truth maintenance [5]. Policy states are organized as a set of justified beliefs among the AMEs. Belief exchange among AMEs uses biology-inspired processes [1].

The remainder of this paper is organized as follows. In section 2, we describe our motivating scenario. Section 3 describes our proposal. Section 4 shows a case study. We compare our research with related works in section 5. Conclusions and future work are finally provided in Section 6.

2 Motivating Scenario: Policies in P2P-based ANM

Different technologies could be employed as an infrastructure of a decentralized ANM system. An interesting possibility is using P2P overlays, which incorporate characteristics of P2P-based network management into ANM systems, such as the support for collaboration in management tasks and robustness in connectivity of management

entities [4].

A biological analogy that can be used is the coherent and stable behavior from cells that form tissues of complex multicellular organisms. Tissues present fault tolerance features in their operation and communication strategies (i.e., cell signalling). In an analogous way, overlay operation could improve service availability replicating messages and operations among different peers.

There are some initiatives investigating P2P-based ANM. In these initiatives, peers have some properties found in AMEs and peer groups have some properties found in AMDs. For sake of simplicity, we present one initiative as an example of P2P-based ANM, as following.

The Madeira platform [3] is an approach to ANM that uses the concept of Adaptive Management Components (AMC). AMCs can communicate with other AMCs running on other managed elements through P2P overlay communication services. AMCs form management with cluster heads, which manages states inside the cluster.

Policy states in different peers of an P2P-based ANM system can become inconsistent in overlay operation due to faults or lack of synchronization. This scenario is similar to those faced in different Multi-Agent Systems (MAS). Thus, techniques from MAS can be used to improve the consistency of policy states in a decentralized ANM system.

3 Proposal

Decentralized ANM systems use several management knowledge bases in their operation (e.g., policies) distributed among the AMEs. A *management knowledge base* is a group of *management data*, which are management information described in a defined form.

The proposed mechanism to improve the consistency of management information in decentralized ANM systems is based on maintenance and exchange of beliefs about management information by AMEs. This mechanism is implemented through a consistency maintenance module that runs in each AME, inspired by *multi-agent truth maintenance* [5]. No other work has attempted to incorporate multi-agent truth maintenance features in a network management system.

The model used for consistency of shared management information is non-deterministic; in other words, it uses a "weak" notion of consistency for scalability and robustness issues [10]. Changes in a belief must be replicated for every AME that shares this belief, so that consistency is enabled among the AMEs that received the same changes.

There are different approaches to multi-agent truth maintenance features, and, in the present work, we used concepts borrowed from distributed *justification-based* TMS. In a justification-based TMS, a datum is believed when it has valid justifications. We are aware of only one work in policy systems that uses TMS technology, as a mono-agent dependency-tracking mechanism [8].

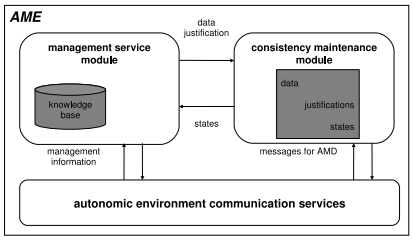

Figure 1. AME architecture

The associated states of a datum are "in" (believed) or "out" (disbelieved), according to its justifications [5]. These justification can be internally generated or received. Thus, the "in" state can assume two additional states: "internal", where the datum has only valid internal justifications, and "external", where the datum has at least one external justification. A datum is labeled "out" when it has no justification associated [5].

In our proposal, some hypothesis are assumed as follows. Management tasks can be performed completely by each one of AMEs. There is support for group organization (i.e., AMD) through offered management services. Thus, AMEs that offer a specific service are organized into an AMD.

3.1 Architecture

The consistency maintenance module is responsible for registration of data and their respective justifications to define AME beliefs about management data. If necessary, the consistency maintenance module uses the autonomic environment communication services to exchange messages. Figure 1 shows the relation between the consistency maintenance module, management service modules, and autonomic environment communication services.

In policy systems, the software component that handles policy processing is in charge of informing changes in justifications, acting as a management service module. When the presence of a justification is modified, the consistency maintenance module performs the following steps: unlabels management datum, includes (or removes) the presence of justification and labels datum again according to new restrictions. The autonomic environment communication services are used to spread changes to AMD.

The consistency maintenance module is offered through a simple interface that management service modules must implement. The number of supported operations is restricted to ease the implementation of management service modules.

3.2 Communication

The consistency maintenance module handles the message exchange through autonomic environment communication services. In this process, requests are adapted in messages to be spread in AMD and vice-versa. The methods used for message exchange inside the AMD are modeled using concepts from biology-inspired distributed computing models [1].

The initial strategy for spreading messages in the AMD is replication. The message exchange replicates changes in data and justifications among the participating entities (i.e., AMEs of a specific AMD). The handling of replicated messages is performed through a selective message discard.

3.3 Evaluation

Our proposal can be evaluated in several different ways. In order to perform a preliminary evaluation, some simulation experiments were made. Decentralized ANM is modelled as P2P-based ANM, thus, the AMEs are simulated as peers and AMDs as peer groups. The simulation experiments were implemented in Java using *PeerSim* [6], an open source event-based simulator of P2P systems.

The simulation results are an evaluation of scalability and robustness claims of our proposal. Each experiment was conducted at least 10 times. Besides, message delay was controlled, varying from 0 (i.e., no delay) to 200 (i.e., twice the cycle length). The Preliminary system version used has the ability to simulate failures in peers and message exchange, and the overlay is built randomly.

In the first experiment, we varied the number of peers of the peer group and we measured the number of messages exchanged to spread justifications change. We consider the number of transmitted messages as indicative of network load. Network load grows linearly with the number of participating peers, thus we can infer the behavior trend of peer groups with larger number of participating peers. However, we do not expect this situation in P2P-based ANM systems. This experiment shows that our system behaves like we expected, without stability and convergence problems.

In the next experiment, we determined the influence of message loss on the dissemination of a justification change. In this experiment, we varied the message loss probability. As can be seen from the results in Figure 2, high loss probabilities do lead to less consistency in peer group, but, even with a few participating peers, the percentage of coherent peers is substantial. Besides, more participating peers in peer group decrease the influence of loss probability.

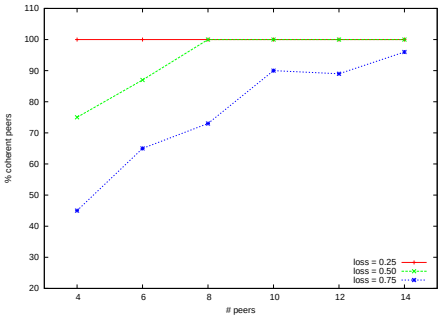

Figure 2. Coherent peers after a justification change

4 Case Study

The case study presented refers to maintenance of consistency of policy states in a P2P-based ANM system. According to our proposal, policy activation is a management data and the event and conditions are justifications in consistency maintenance module. The policy could be in two different states, “in” or “out”. In addition, the state “in” has two options: ”internal” (every justification was generated internally) and “external” (some justification was received through replicated messages).

We describe the policy used in this case study using *Ponder2* [9]. The code below shows shows an excerpt from the XML encoding of an obligation policy that will respond to events of type */event/humanResourcesProcedureEvent*. In the example, Ponder2 checks whether the source address is 10.0.0.1/24, and the current time is later than 18:00 and, if the conditions are satisfied, the policy invokes an action on the router R1 to reserve 10% of bandwidth in “QoS1” profile.

```
...
<create type="obligation"
event= "/event/humanResourcesProcedureEvent">
  <condition>
   <and>
    <eq>!source_address;<!-- -->10.0.0.1/24</eq>
    <gt>!daytime;<!-- -->18:00</gt>
   </and>
  </condition>
  <action>
   <use name="/routers/R1">
    <modify profile="QoS1" value="10%"/>
...
```

The consistency maintenance module receives the datum “*AdjustQoSPolicy*” and its justification list, composed by “*humanResourcesProcedureEvent received*”, “*source address matched*” and “*daytime matched*”. Initially, the datum has the state “out”, afterwards, this state can be changed

when the policy module inform the presence of a justification or a message is received. We are primarily concerned with two situations that can occur: sent events are not received by some peers, and, due to lack of synchronization, some peers disagree about a temporal condition.

The reception process of a specific event and the evaluation of a temporal condition can be used as examples to illustrate the operation of consistency maintenance module in a P2P-based ANM system. In our proposal, after reception of an event or a change in a temporal condition, policy module informs the belief change to the consistency maintenance module. If the new belief implies a justification change, a message will be sent to the peer group informing the change. When receiving this message, consistency maintenance module of other peers checks whether the received justification change is consistent with their knowledge base. If it is not, the consistency maintenance module updates the justification and verifies if this update implies other changes.

In both cases, even a peer that has not produced some of its justifications, but that had received a justification change message, will be coherent with other peers of the peer group. Thus, this policy state keeps consistent and the P2P-based ANM system show a coordinated behavior.

5 Related Work

5.1 Truth Maintenance Systems

Truth Maintenance Systems (TMS) determine the current set of beliefs from the current set of reasons, and update the current set of beliefs in accord with new reasons in mono-agent systems. These systems were extended for multi-agent versions [5].

TMS technology is not well known outside the artificial intelligence community. Nevertheless, TMSs are used in different contexts (e.g., plan adaptation and repair). We are aware of only one work that uses TMS in policy systems [8]. In this work, TMS is used as a dependency-tracking mechanism to provide explanations for policies.

5.2 Consistency of shared information in distributed systems

The consistency of shared management information has an important role in the coherent behavior in a decentralized ANM systems. The service presented, in spite of not being developed for decentralized ANM systems, has analogous characteristics with our proposal.

Astrolabe [10] is a distributed information management service. *Astrolabe* is implemented using a P2P overlay, where every peer run an *Astrolabe* agent (i.e., in a MAS fashion). *Astrolabe* was developed primarily for queries to simple data and for application with a profile oriented for read operations

6 Conclusions and Future Work

In this paper we have defined a module to improve the consistency maintenance of policy states in decentralized ANM systems through the utilization of multi-agent truth maintenance. We have also presented evaluations of this proposal through simulations. In addition, we have described a case study using obligation policies to show the possibilities of our proposal.

Although we believe the proposal shows a relevant mechanism to ANM systems, other features can further improve our proposal, such as the utilization of a language suitable for machine learning in justifications. It is also important to evaluate other operational details, and, currently, we are working on new simulations and an implementation.

References

[1] O. Babaoglu, G. Canright, A. Deutsch, G. Di Caro, F. Ducatelle, L. Gambardella, N. Ganguly, M. Jelasity, R. Montemanni, A. Montresor, et al. Design patterns from biology for distributed computing. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 1(1):26–66, 2006.

[2] EMANICS. *Activities - Work Package 9 - Autonomic Management*. http://emanics.org/content/view/63/109/, 2009.

[3] L. Fallon, D. Parker, M. Zach, M. Leitner, and S. Collins. Self-forming Network Management Topologies in the Madeira Management System. *Lecture Notes in Computer Science*, 4543:61, 2007.

[4] L. Z. Granville, D. M. da Rosa, A. Panisson, C. Melchiors, M. J. B. Almeida, and L. M. R. Tarouco. Managing computer networks using peer-to-peer technologies. *IEEE Communications Magazine*, 43(10):62–68, 2005.

[5] M. N. Huhns and D. M. Bridgeland. Multiagent truth maintenance. *Systems, Man and Cybernetics, IEEE Transactions on*, 21(6):1437–1445, 1991.

[6] Jelasity, M. and Montresor, A. and Jesi, G. and Voulgaris, S. *The Peersim Simulator*. http://peersim.sf.net, 2008.

[7] B. Jennings, S. Van Der Meer, S. Balasubramaniam, D. Botvich, M. O. Foghlu, W. Donnelly, and J. Strassner. Towards autonomic management of communications networks. *IEEE Communications Magazine*, 45(10):112–121, 2007.

[8] L. Kagal, C. Hanson, and D. Weitzner. Using Dependency Tracking to Provide Explanations for Policy Management. In *Policies for Distributed Systems and Networks, 2008. POLICY 2008. IEEE Workshop on*, pages 54–61, 2008.

[9] Ponder2 Documentation. http://www.ponder2.net/.

[10] R. Van Renesse, K. Birman, and W. Vogels. Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. *ACM Transactions on Computer Systems (TOCS)*, 21(2):164–206, 2003.

APÊNDICE B ARTIGO PUBLICADO – DSOM 2009

Neste apêndice é apresentado o artigo “Consistency of States of Management Data in P2P-based Autonomic Network Management”, desenvolvido durante a execução do trabalho descrito na presente dissertação. O artigo apresenta uma versão preliminar da proposta de introdução de funcionalidades de manutenção da verdade multiagente em sistemas Gerenciamento Autonômico de Redes baseados em P2P. Além disso, o artigo apresenta uma estratégia de comunicação (replicação livre) para prover suporte a essas funcionalidades. Um estudo de caso relativo ao gerenciamento de falhas em enlaces de rede Ethernet em provedores de serviços é descrito para mostrar a viabilidade dessa proposta. Finalmente, experimentos simulados são realizados para avaliar a estratégia de comunicação apresentada.

- **Título**: *Consistency of States of Management Data in P2P-based Autonomic Network Management*
- **Evento**: *20th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM 2009)*
- **URL**: http://www.manweek.org/2009/dsom/
- **Data**: De 27 a 28 de outubro de 2009
- **Local**: Veneza, Itália

Consistency of States of Management Data in P2P-based Autonomic Network Management

Jéferson Campos Nobre, Lisandro Zambenedetti Granville

Institute of Informatics – Federal University of Rio Grande do Sul
Av. Bento Gonçalves, 9500 - Porto Alegre, RS - Brazil
{jcnobre,granville}@inf.ufrgs.br

Abstract. Autonomic network management is a vision that brings autonomic computing principles to network management. Besides, it is necessary some level of decentralization to enable broad autonomic capabilities. An interesting alternative of infrastructure for this union is the utilization of peer-to-peer (P2P) overlays. However, the consistency of states of management data among peers is an important challenge. Traditional mechanisms to maintain consistency of these states are supported by some centralization which wastes some desirable properties of P2P facilities. In contrast to these mechanisms, we propose a distributed, scalable and robust mechanism to maintain the consistency of states of management data. In this paper we introduce multi-agent truth maintenance features in P2P-based autonomic network management as a mechanism to bring consistency maintenance of these states. We developed a model of a P2P-based autonomic network management system on Peersim to perform simulation experiments. Besides, the utilization of P2P-based autonomic network management systems in access networks is presented as a case study.

1 Introduction

The increasing complexity of computer networks requires sophisticated solutions to manage the underlying communication infrastructure and help network human administrators in their daily tasks [1]. The application of autonomic computing (AC) principles in network management, normally refereed as *Autonomic Network Management* (ANM), has been proposed as a way to address some demands faced by traditional network management, such as controlling highly dynamic environments like ad-hoc and peer-to-peer (P2P) networks [2]. ANM systems increase the efficiency of network human administrators by decreasing the number of manual interventions. This efficiency improvement is done by ANM systems through automation and/or optimization of some operational details of management tasks, such as fault handling and performance management.

ANM systems can be deployed using different models of management distribution, from totally centralized models up to highly decentralized ones. Today, there are no clear evidences that could link the quality of autonomic actions of an ANM system with the distribution model adopted. Some authors, however, claim that some level of decentralization is required to achieve a more adequate ANM [3]. In this scenario, the

typical design of decentralized ANM systems is based on a set of *Autonomic Management Elements* (AME) that execute management tasks and interact with one another to form an *Autonomic Management Domain* (AMD). Multiples AMDs could be integrated to form an *Autonomic Management Environment*.

Different technologies can be employed to provide an infrastructure for decentralized ANM systems. An interesting possibility is using peer-to-peer (P2P) overlays, which incorporate characteristics of P2P networks into ANM systems, such as the support for collaborative management, robustness in connecting management entities, and load balancing of management tasks [4]. There are some initiatives investigating P2P-based ANM [5] [6] and, in these initiatives, peers have some properties found in AMEs. Besides, peer with similar properties (e.g., managing the same devices) can be organized into groups. These peer groups have some properties found in AMDs.

A managed element (e.g., a network router) can be controlled by one or multiple peers in P2P-based ANM system, for example, for the purpose of management robustness. In this case, the status of the managed element becomes a distributed, replicated information among the controlling peers. Inconsistencies in this information could lead to an improper operation of the ANM system. For example, the status of a router's link (e.g., ethernet interface) can be advertised differently by the controlling peers to external requesters (e.g., management station). This inconsistency can be caused by faults in network (e.g., losses in messages exchanged among peers) and computational resources (e.g., incorrect operation of peers). Besides, inconsistency in management data can occur even in the regular operation of P2P-based ANM system, due to lack of information synchronization among peers. This information synchronization is a challenge in unrealible asynchrounous distributed systems, such as an unstructured P2P overlay.

Mechanisms to support consistency of management information in current P2P-based ANM systems are, in general, still supported by some centralization through, for instance, external repositories [7] or *super peers* [5] [6], missing opportunities of unstructured P2P facilities. This centralization complicates the achievement of good scalability and robustness features, thus, in spite of being P2P-based ANM systems, these examples show undesirable characteristics of client-server systems.

In this paper, we introduce *Multi-Agent Truth Maintenance* features [8] in the P2P-based ANM to improve the consistency of states of management data. The introduction of these features maintains desirable scalability and robustness characteristics of P2P-based ANM. It is also proposed a communication strategy for message exchange among peers to support this truth maintenance. This strategy uses biology-inspired processes (e.g., replication), which have well know scalability and robustness features [9].

The remainder of this paper is organized as follows. Section 2 discusses related works. Section 3 describes our proposal and its implicit concepts. Section 4 shows a case study. Section 5 shows evaluations of our proposal. Conclusions and future work are finally provided in Section 6.

2 Background

In this section we first discuss about the current state-of-the-art on P2P-based autonomic network management. Afterwards, the main concepts behind truth maintenance

systems are presented. Finally, some important services related to the maintenance of consistency of shared information in distributed applications are discussed.

2.1 P2P-Based Autonomic Network Management

The utilization of P2P overlays is identified as an emerging approaches for Autonomic Network Management (ANM). This can be specially observed in the increasing number of research work towards this utilization [10].

PBMAN [5] merge traditional PBNM with P2P overlays to autonomically manage Ambient Networks (AN). PBMAN enables scalable mechanisms for network composition inside the AN, as well as policies distribution and retrieval. Through this approach it is possible to establish policies to manage devices or services. PBMAN is structured using super peers, in an hierarchical architecture. These super peers are responsible for consistency of states of management data, among other functions, in each hierarchical level.

The Madeira platform [6] is an approach to ANM that uses the concept of Adaptive Management Components (AMC), which are containers that run on managed elements. AMCs can communicate with other AMCs running on other managed elements through P2P communication services. AMCs form management clusters with super peers acting as cluster heads. These super peers are responsible for consistency of states of management data, among other functions.

ManP2P [11] is a P2P-based network management system that is evolving to an autonomic conception through the implementation of autonomic modules in peers [7]. ManP2P is partially inspired by the Management by Delegation (MbD) model and based on a service-oriented approach. There is no internal mechanism for consistency of state of management data, thus, authors propose the utilization of external repositories.

Despite many improvements brought by the utilization of P2P-based ANM systems, there are still issues to be addressed. The consistency of state of management data among the peers is usually addressed with some centralization, using super peers [5] [6] or external repositories [7], which misses opportunities of P2P overlays (e.g, robustness).

2.2 Truth Maintenance Systems

Truth-Maintenance Systems (TMS) were proposed to keep the integrity of Knowledge Bases (KB). The origin of these systems was proposed in the 1970s, for resolutions in mono-agent systems [12]. A TMS keeps track of logical structure of the set of beliefs of agents. A belief is a member of the current set of beliefs if it has valid reasons.

TMSs provide considerable power using few computational resources [13]. Thus, although not being well known outside artificial intelligence community, TMSs are used in different contexts, such as policy systems [13] and network management systems [14].

TMSs have been extended for Multi-Agent System (MAS) versions, *Distributed Truth-Maintenance Systems* (DTMS) [8]. In a MAS, agents must be able to maintain the integrity of their KBs, despite message exchange with other agents. This maintenace of

integrity can be done by a multi-agent TMS. In an analogous way, during the operation of P2P-based ANM system, peers must be able to maintain the integrity of states of management data, despite message exchange with other peers. This similarity indicates the use of multi-agent TMS in P2P-based ANM as an interesting possibility [14].

2.3 Services for consistency of shared information in distributed systems

Services for consistency of shared information can be used as a basic building block for distributed applications. A P2P-based ANM system, as a distributed application, can appropriate some good characteristics found in these systems.

ZooKeeper [15] is a coordination service for distributed applications. It exposes a simple API that distributed applications can be built upon to implement higher level services for synchronization, data diffusion, and publish-subscribe schemes. *ZooKeeper* use distributed server databases for read operations, however, write operations use a "leader" server (i.e., centralized database) to assure the consistency of the database.

Astrolabe [16] is a distributed information management service. It works locating and collecting the status of a set of servers and reporting summaries of this information. *Astrolabe* is implemented using a P2P overlay, where every peer run an *Astrolabe* agent (i.e., in a MAS fashion). However, *Astrolabe* was developed primarily using simple data models. Besides, its operation is aimed at read-oriented applications.

Scalable Distributed Information Management System (SDIMS) [17] is a service to aggregate information about large-scale network systems. The service is built using ideas from *Astrolabe* [16] and Distributed Hash Tables (DHT). However, as in most DHT approaches, consistency and replication issues are a known challenge.

The presented efforts show interesting characteristics for consistency of shared information in distributed systems. However, these efforts have vulnerabilities which make them not appropriate for P2P-based ANM systems, such as centralization [15], simple data models [16], and replication issues [17].

The consistency of states of management data is still an issue to be adressed in P2P-based ANM systems. It is necessary to maintain this consistency keeping scalability and robustness features of P2P overlays. The maintenance of integrity of KBs among agents brought by multi-agent TMS seems to be a valid direction to introduce this consistency maintenance [14]. Besides, it is interesting to bring characteristicas of services for consistency of shared information.

3 Proposal

In P2P-based Autonomic Network Management (ANM) systems, peers must share management data. In this work, *management datum* is defined as a management information described in a defined form (i.e., using a specific language). Besides, in these systems, management data must allow their use in automation and/or optimization procedures. It is also expected that sources of management data (e.g., highly dynamic environments) impose challenges to the ANM system. Despite these requirements, it is necessary to avoid potencial inconsistencies in state of management data among peers.

Our proposal is aimed at meeting requirements of consistency of states of management data in a P2P-based ANM system. The proposed mechanism introduces *multi-agent truth maintenance* [8] features through a *consistency maintenance module* that runs in each peer. As far as we are aware of, the only study that incorporates multi-agent truth maintenance features in a network management system was carried out by *Nobre and Granville* [14], and only some results were published so far.

Multi-agent truth maintenance is a multi-agent extension to Truth-Maintenance Systems (TMS). The goal of TMS utilization is to keep the integrity of knowledge bases. In this work, a *knowledge base* is a group of integrated management data. In a multi-agent TMS, there are multiple agents and each one has its own TMS. TMSs keep integrity performing belief revision and exchange in a set of beliefs. A belief is a member of the current set of beliefs if it has valid reasons.

The exchange of beliefs about management data is done asynchronously and we do not consider the message exchange to be reliable. Unfortunately, it is well known that the utilization of asynchronous unreliable distributed systems imposes challenges to achieve consistency in shared data. Thus, the consistency model used is non-deterministic, in other words, it uses a "weak" notion of consistency. This model is adopted for scalability, robustness, and update dissemination issues. Given a belief X that depends on some other belief Y, when an update is made to Y, it is eventually reflected in X. Some authors call this notion as "eventual consistency" [16]

3.1 Justifications for Management Data

Multi-agent TMS is a kind of *justification-based* TMS. In a justification-based TMS, a datum is believed when it has valid justifications (i.e., valid reasons). This belief can be shared by different peers (which play the agent role) in a P2P-based ANM system (which aggregates some MAS characteristics). Thus, justifications improve th alignment of peers with system-wide objectives (i.e., objectives of the P2P-based ANM system). The datum and its list of possible justifications must be provided by network human operators or expert systems for the P2P-based ANM system.

The associated states of a datum are "in" (believed) or "out" (disbelieved), according to its justifications. These justification can be generated by processes inside the peer or received through the P2P overlay communication services. Thus, the "in" state can assume two additional states: "internal", where the datum has only valid internal justifications, and "external", where the datum has some valid external justification (provided by other peer). A datum is labeled "out" when it lacks, at least, one of its associated justifications [8].

For instance, the activation (belief) of a QoS policy (datum) can be justified by a network human administrator command (justification) and an asynchronous signal from a managed device (justification). The code below shows a possible internal representation of this datum and its justifications. In the example, each datum or justification ("just" in the code) is represented with 2 fields, "name" and "desc" (description). The last line of the code defines that if the justifications "adm_cmd" and "async_sig" are present, the datum "qos_pol" is believed.

```
datum:  name qos_pol
```

```
        desc QoS policy
just:   name adm_cmd
        desc network human administrator command
just:   name async_sig
        desc asynchronous signal from a managed device
tms:    qos_pol (adm_cmd async_sig)
```

3.2 Architecture of Peers

Peers are commonly viewed as composed by one or more *management service modules*. *Management service modules* perform regular management tasks (e.g., collecting statistics) in each peer, and, in these tasks, modules produce management data, building *management knowledge bases*.

We introduce the *consistency maintenance module* to registrate the set of belief about management data in each peer. This module works associating management data and their respective justifications. When there is a belief change (i.e., justitication change), the consistency maintenance module uses the P2P overlay communication services to spread the change. Figure 1 shows the relation between the consistency maintenance module, management service modules, and P2P overlay communication services.

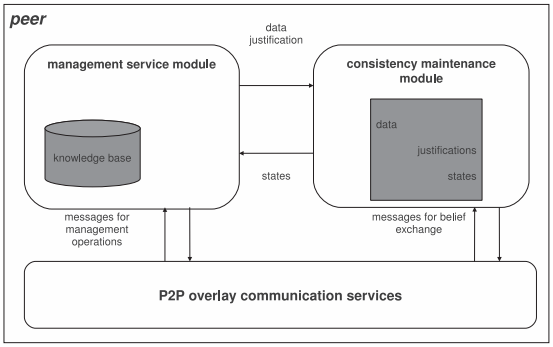

Fig. 1. Peer architecture

The management service modules should inform consistency maintenance module about their internal beliefs of management data. The management service modules are also responsible for querying and requiring services from the consistency maintenance module, possibly updating their internal beliefs. These beliefs are exchanged through

justification using a simple interface that management service modules and consistency maintenance module must use.

When the presence of a justification is modified, the consistency maintenance module performs the following steps: unlabels management datum, includes (or removes) the presence of justification and labels datum again according to new restrictions. The P2P overlay communication services are used to spread changes, which can change beliefs of other peers.

It is important to stress that there is only one consistency maintenance module inside a peer, thus, it is not specific of a management service module. Therefore, every management service module in a peer interacts with the same consistency maintenance module. This fact can be explored for the integration of different management services. For instance, a policy processing module, a fault handling module, and a configuration management module (possibly using different languages for representing management data) could be integrated by the consistency maintenance module through justifications.

3.3 Communication within a peer group

The consistency maintenance module handles the message exchange through P2P overlay communication services. In this process, requests are adapted in messages to be spread among peers and vice-versa. The P2P overlay is modelled as an unstructured overlay networks, thus there is no relation between the information stored at a peer and its position in the overlay topology.

We use the premise that there is support for group organization (i.e., peer groups) through management services modules. Thus, peers that have a specific management service module are organized into a group (without human intervention) and peers can participate of several groups accordingly to modules that they have.

The methods used for message exchange inside the peer group are modeled using concepts from biology-inspired distributed computing models [9]. Among these models, proliferation-based ones are an interesting choice for communication requirements of our proposal. All peers in the peer group run exactly the same communication algorithm, which can be initiated from any peer in the peer group.

We have chosen *replication* as the initial proliferation mechanism in the peer group. This mechanism can support a number of different strategies [9]. In our proposal, peers spread messages to replicate changes in justifications among the participating entities (i.e., peers of a specific peer group). This unbridled replication is restricted to peer group, fulfilling the criterion of robustness and controlling the number of messages within the P2P overlay.

4 Case Study

The case study presented is an illustration of the collaborative fault management of links in access networks through failure notification sent by devices and human knowledge about these notifications. The integration of these information (failure notification in addition to human knowledge) produces a management datum, which can assume

different states. This datum can be used against a Service-Level Agreements (SLAs) to support or clarify service level claims.

Among access network technologies in metropolitan networks, Ethernet is one of most interesting and promising choice, thus, we choose it to build our case study. In this context, an access network link is an *Ethernet Virtual Connection* (EVC) [18]. Fault management in this link is done through *Alarm Indication Signal* (AIS) messages [18]. These messages are triggered when a failure between two nodes occurs. Thus, AIS messages provide asynchronous notification to other elements in the network that there is a fault in the Ethernet network. The efforts to manage layer 2 Ethernet service must consider an overlayed IP infrastructure [19].

A management service module collects AIS messages and another module collects information from human administrators. The consistency maintenance module is responsible to integrate the information from both management service modules and maintain the consistency of the state of management datum in the peer group.

The *link failure detected* (datum) is justified by human administrator commands from both domains, *network operator detection* and *network consumer detection*, and a *device notification received* (AIS message). The code below shows the representation of this datum and its justifications. These justifications are provided by management service modules and kept inside the peer group that offers this management service.

```
datum:  name link_fail_det
        desc link failure detection
just:   name net_opt_det
        desc network operator detection
just:   name net_con_det
        desc network consumer detection
just:   name ais_msg
        desc ais message
tms:    link_fail_det (net_opt_det net_con_det ais_msg)
```

Initially, the datum has the “out” state, since the justifications are not present. This state can be changed after the verification of new beliefs (i.e., justification changes) by consistency maintenance module. These beliefs are informed by management service modules or received as a message in the peer group. If every justification of the datum is present, the state changes to “in”. This “in” state assume two options according the sources of justifications: “internal” (every justification was generated internally) and “external” (some valid external justification).

For instance, human administrator commands (network operator detection and network consumer detection) can be generated internally and the presence of an AIS message can be received as a justification change message. In this example, the “in” state assumes the option “external”. The code below shows the answer from the consistency maintenance module in this situation.

```
tms:    link_fail_det:external (net_opt_det:mod net_con_det:mod ais_msg:msg)
```

This management operation is traditionally performed through separate centralized systems, a network management system (collecting notifications from devices), and a trouble ticket system (collecting information from human administrators). The traditional procedure brings concerns in scalability and robustness and imposes difficulties

in the integration of the information. Besides, justifications can be use to provide explanations to the user [13], improving the understanding of management data.

5 Evaluation

The evaluation of our proposal can be performed in different ways. To enable a fully controlled environment for the evaluation, we chose to develop some simulation experiments. In these experiments, we present simulation results that support our scalability and robustness claims. Scalability and robustness are some of the most important motivations for using decentralization in the infrastructure of different systems [20] [9], such as P2P facilities.

As previously stated, we expect that the introduction of multi-agent TMS features keeps decentralization properties of a P2P-based ANM system, maintaining each peer as an independent and self-sustainable entity. As many systems have demonstrated, a system that does not share resources can scale almost infinitely simply by adding constitutive elements (e.g, peers in a P2P-based ANM system). Besides, maintaining the independence of each peer, single points of failure are eliminated.

The simulation experiments were implemented in Java using *PeerSim* [21], an open source event-based simulator of P2P systems. The preliminary system version used has the ability to simulate failures in peers and message exchange, and the overlay is built randomly. The experiments use a simple model of transport layer that can emulate some characteristics, such as loss and delay probabilities. All peers in peer groups run exactly the same algorithm.

In the experiments, we varied the number of peers of the peer group from 4 to 14 (we do not expect large peer groups in P2P-based ANM systems). Besides, these peer group sizes seems reasonable for the case study provided in Section 4. In addition, a peer is chosen randomly as the primary source of changes to not affect measurements and message delay is controlled. Each experiment was conducted at least 10 times. In the experiments, the variance observed was low.

In the first experiment, it is measured the number of messages exchanged to spread justification changes in the peer group. This number must be considered as an important cost of the peer group operation, thus, it is important for scalability analysis. Besides, we consider the number of transmitted messages as indicative of network load. In this experiment there were no faults in peers or in message exchange. We show the results in Figure 2.

Our proposal shows acceptable scalability characteristics on number of exchanged messages, since this operation (message exchange) is restricted to each peer group. The experiment shows that our system behaves like we expected, without stability and convergence problems. Network load grows linearly with the number of participating peers, thus we can infer the behavior trend of peer groups with larger number of participating peers. Of course, an efficient operation of large peer groups needs modifications in communication strategies, such as the utilization of gossip-based protocols.

In the second experiment, we determined the influence of message loss on the dissemination of a justification change. In this experiment, we varied the message loss probability with following values: 25%, 50%, and 75% (respectively, 0.25, 0.5, and 0.75

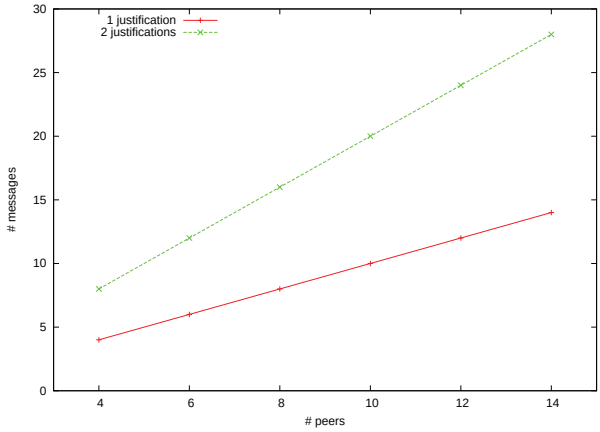

Fig. 2. Message exchange due to justification changes

as indicated in Figure 3). Using our case study, we would probably observe such message loss (specially 75%) due to faulty or overloaded network equipment (e.g., ethernet interfaces) and/or network links (e.g., ethernet lines). Since our case study is aimed at fault management (considering an overlayed IP infrastructure), our system must behave acceptably even in bad network conditions. In Figure 3, we show the average percentage of coherent (and correct) peers after message exchange to cease.

The experiment shows the influence of message loss in the replication process. As can be seen from the results in Figure 3, high loss probabilities do lead to less consistency in peer group, but, even with a few participating peers, the percentage of coherent peers is substantial. Besides, more participating peers in peer group decrease the influence of loss probability.

The results show some fault-tolerance features, since the peer group operation is not highly sensitive to peer crashes and message losses. But an increase in number of peers also leads to an increase in the number of exchanged messages, so the robustness advantages come at some cost. However, messages are exchanged only within the peer group, and a high number of peers in a peer group is not expected. Thus, the number of exchanged messages does not impose a issue in scalability.

6 Conclusions and Future Work

The support of new demands faced by traditional network management is a key research issue in network management area. One of these demands is the support for collaborative management tasks over access network links. In order to enable efficiently these tasks, it is necessary some decentralization and automation features. These features can

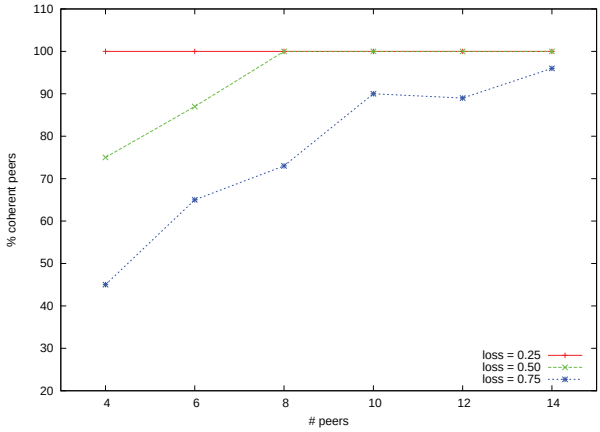

Fig. 3. Coherent peers after a justification change

be supported by P2P-based autonomic network management (ANM) systems. However, the consistency of states of management data imposes challenges for these systems.

In this paper we have introduced multi-agent TMS features to improve the maintenance of consistency of states of management data in P2P-based ANM systems. Our proposal aims at the integration of data used by the entities that form these systems (i.e., peers), through the utilization of justifications. We have also presented evaluations of this proposal through simulation experiments. In addition, we have described a case study of fault management in access networks to show the possibilities of our proposal.

Although the proposal shows good results in evaluations performed until the present moment, it is necessary to evaluate more complicated cases, in number of peers and peer groups, and in the participation of a peer in different peer groups. We are also looking at additional settings that could lead to important effects, such as network partitions. Thus, we are currently pursuing new experiments with *PeerSim*.

References

1. Kind, A., Dimitropoulos, X., Denazis, S., Claise, B.: Advanced network monitoring brings life to the awareness plane. IEEE Communications Magazine **46**(10) (2008) 140–146
2. Pras, A., Schoenwaelder, J., Burgess, M., Festor, O., Perez, G.M., Stadler, R., Stiller, B.: Key research challenges in network management. IEEE communications magazine **45**(10) (October 2007) 104–110
3. Jennings, B., Van Der Meer, S., Balasubramaniam, S., Botvich, D., Foghlu, M.O., Donnelly, W., Strassner, J.: Towards autonomic management of communications networks. IEEE Communications Magazine **45**(10) (2007) 112–121

4. Granville, L.Z., da Rosa, D.M., Panisson, A., Melchiors, C., Almeida, M.J.B., Tarouco, L.M.R.: Managing computer networks using peer-to-peer technologies. IEEE Communications Magazine **43**(10) (2005) 62–68
5. Kamienski, C., Fidalgo, J., Sadok, D., Lima, J., Pereira, L., Ohlman, B.: PBMAN: A Policy-based Management Framework for Ambient Networks. In: Policies for Distributed Systems and Networks, 2006. POLICY 2006. IEEE Workshop on. (2006) 79–83
6. Fallon, L., Parker, D., Zach, M., Leitner, M., Collins, S.: Self-forming Network Management Topologies in the Madeira Management System. Lecture Notes in Computer Science **4543** (2007) 61
7. Marquezan, C.C., dos Santos, C.R.P., Nobre, J.C., Almeida, M.J.B., Tarouco, L.M.R., Granville, L.Z.: Self-managed services over a p2p-based network management overlay. In: Proceedings. 2nd Latin American Autonomic Computing Symposium (LAACS 2007). (2007)
8. Huhns, M.N., Bridgeland, D.M.: Multiagent truth maintenance. Systems, Man and Cybernetics, IEEE Transactions on **21**(6) (1991) 1437–1445
9. Babaoglu, O., Canright, G., Deutsch, A., Di Caro, G., Ducatelle, F., Gambardella, L., Ganguly, N., Jelasity, M., Montemanni, R., Montresor, A., et al.: Design patterns from biology for distributed computing. ACM Transactions on Autonomous and Adaptive Systems (TAAS) **1**(1) (2006) 26–66
10. EMANICS: Activities - Work Package 9 - Autonomic Management. http://emanics.org/content/view/63/109/ (2009)
11. Panisson, A., da Rosa, D.M., Melchiors, C., Granville, L.Z., Maria, Liane: Designing the Architecture of P2P-Based Network Management Systems. In: ISCC '06: Proceedings of the 11th IEEE Symposium on Computers and Communications, IEEE Computer Society (2006) 69–75
12. Doyle, J.: A truth maintenance system. Computation & intelligence: collected readings (1979) 529–554
13. Kagal, L., Hanson, C., Weitzner, D.: Using Dependency Tracking to Provide Explanations for Policy Management. In: Policies for Distributed Systems and Networks, 2008. POLICY 2008. IEEE Workshop on. (2008) 54–61
14. Nobre, J.C., Granville, L.Z.: Towards consistency of policy states in decentralized autonomic network management. In: Policies for Distributed Systems and Networks, 2009. POLICY 2009. IEEE Workshop on (to appear). (2009)
15. Hunt, P.: ZooKeeper: A Distributed Coordination Service for Distributed Applications. http://wiki.apache.org/hadoop/ZooKeeper (2008)
16. Van Renesse, R., Birman, K., Vogels, W.: Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. ACM Transactions on Computer Systems (TOCS) **21**(2) (2003) 164–206
17. Yalagandula, P., Dahlin, M.: A scalable distributed information management system. ACM SIGCOMM Computer Communication Review **34**(4) (2004) 379–390
18. McFarland, M., Salam, S., Checker, R.: Ethernet oam: key enabler for carrier class metro ethernet services. IEEE Communications Magazine **43**(11) (2005) 152–157
19. Ryoo, J., Song, J., Park, J., Joo, B.: Oam and its performance monitoring mechanisms for carrier ethernet transport networks. Communications Magazine, IEEE **46**(3) (2008) 97–103
20. Mccann, J.A., Huebscher, M.C.: Evaluation Issues in Autonomic Computing. In: Grid and Cooperative Computing – GCC 2004. Volume 3252 of Lecture Notes in Computer Science., Heidelberg, Springer-Berlim (2004) 597–608
21. Jelasity, M. and Montresor, A. and Jesi, G. and Voulgaris, S.: The Peersim Simulator. http://peersim.sf.net (2008)

APÊNDICE C ARTIGO PUBLICADO – NOMS 2010

Neste apêndice é apresentado o artigo “Consistency Maintenance of Policy States in Decentralized Autonomic Network Management”, desenvolvido durante a execução do trabalho descrito na presente dissertação. O artigo apresenta a proposta de introdução de funcionalidades de manutenção da verdade multiagente em sistemas de Gerenciamento Autonômico de Redes descentralizados. Além disso, o artigo apresenta uma estratégia de comunicação (replicação controlada) para prover suporte a essas funcionalidades. Um estudo de caso relativo à manutenção da consistência do estado de ativação de políticas de gerenciamento de rede é descrito para mostrar a viabilidade dessa proposta. Finalmente, experimentos simulados são realizados para avaliar a estratégia de comunicação apresentada.

- **Título**: *Consistency Maintenance of Policy States in Decentralized Autonomic Network Management*
- **Evento**: *12th IEEE/IFIP Network Operations and Management Symposium (NOMS 2010)*
- **URL**: http://www.ieee-noms.org/2010/
- **Data**: De 19 a 23 de abril de 2010
- **Local**: Osaka, Japão

Consistency Maintenance of Policy States in Decentralized Autonomic Network Management

Jéferson Campos Nobre, Lisandro Zambenedetti Granville
Institute of Informatics - Federal University of Rio Grande do Sul
Porto Alegre, RS, Brazil
Email: {jcnobre, granville}@inf.ufrgs.br

***Abstract*—Autonomic network management is a vision that brings autonomic computing principles to network management. In this vision, the use of policies is a key aspect. Besides, it is necessary to add some level of decentralization to enable broad autonomic capabilities. Thus, the elements that build decentralized autonomic network management systems (known as autonomic management elements) must be policy-enabled. However, the consistency of policy states among autonomic management elements is an important challenge. Traditional mechanisms to maintain consistency of these states are supported by some centralization which wastes some desirable properties of decentralization. In contrast to these mechanisms, we propose a distributed, scalable and robust mechanism to maintain the consistency of policy states. In this paper we introduce multi-agent truth maintenance features in decentralized autonomic network management as a mechanism to bring consistency maintenance of policy states. We developed a model of a decentralized autonomic network management system on Peersim to perform simulation experiments. Besides, the utilization of policies in P2P-based autonomic network management systems is presented as case study.**

I. Introduction

The increasing complexity of computer networks requires sophisticated solutions to manage the underlying communication infrastructure and help network human administrators in their daily tasks [1]. The application of autonomic computing (AC) principles in network management, normally referred as *Autonomic Network Management* (ANM), has been proposed as a way to address some demands faced by traditional network management, such as controlling highly dynamic environments such as ad-hoc networks [2]. ANM systems increase the efficiency of network human administrators, by decreasing the number of manual interventions. This efficiency improvement is done by ANM systems through automation and/or optimization of some operational details of management tasks (e.g., fault handling) [1].

ANM systems can be implemented through different models of management distribution, from totally centralized management to highly decentralized. Today, there is no clear evidence that could link the quality of autonomic actions of an ANM system with the distribution model adopted. Some authors, however, claim that some level of decentralization is required to achieve a more adequate ANM [1]. In this scenario, the typical design of decentralized ANM systems is based on a set of *Autonomic Management Elements* (AME) that execute management tasks and interact to form an *Autonomic Management Domain* (AMD). Multiple AMDs could be integrated to form an *Autonomic Management Environment* [1].

Policies play a key role in ANM systems [3] [1]. Policy-Based Network Management (PBMN) is proving to be a feasible approach for network management and represents an enabling technology to autonomic management capabilities as well [3]. For example, policies can be used to automate the decision making process. In a policy-enabled decentralized ANM system, AMEs must be policy-aware and take part in management decisions. These decisions are processed through the analysis of requests against the network policy. After that, a final decision is returned to the decision requester, which may be a network device (e.g., IntServ routers), a brokering entity (e.g., DiffServ bandwidth brokers), or a human operator.

A managed element can be controlled by one or multiple AMEs (e.g., for robustness purposes). In the latter case, policies states (i.e., management decisions) become distributed information. Inconsistency in this information could lead to a improper operation of the ANM system. For instance, the state of a QoS obligation policy (i.e., if the policy is enabled or not) can be advertised differently by AMEs to a specific IntServ router. This failure could be caused by faults in network and computational resources or even due to lack of synchronization among AMEs.

Mechanisms to support consistency of policy states in current decentralized ANM systems are, in general, still supported by some centralization [4] [5], which wastes some decentralization properties (e.g., scalability). In other approaches, the mechanism to distribute policy states is not clearly stated [1]. An effective distribution of policy states demand a mechanism to improve their consistency, avoiding incoherent advertisements. Besides, the communication strategy of the mechanism must be scalable and robust.

In this paper, we propose a consistency maintenance module in each AME to improve the consistency of policy states in decentralized ANM systems. The module introduces features inspired by multi-agent truth maintenance [6], a concept borrowed from Multi-Agent Systems (MAS). Policy states are organized as a set of justified beliefs among the AMEs. A communication strategy within an AMD to support belief exchange among AMEs is also proposed. This strategy uses biology-inspired processes (e.g., replication), which have well know scalability and robustness features [7].

The remainder of this paper is organized as follows. In

section 2, we describe our motivating scenario. Section 3 describes our proposal and its implicit concepts. Section 4 shows a case study. We compare our research with related work in section 5. Conclusions and future work are finally provided in Section 6.

II. Motivating Scenario: Policies in P2P-based ANM

Different technologies could be employed as an infrastructure of a decentralized ANM system. An interesting possibility is using P2P overlays, which incorporate characteristics of P2P-based network management into ANM systems, such as the support for collaboration in management tasks, robustness in connectivity of management entities, and load balance of management tasks [8].

Several aspects of P2P overlays can be investigated in order to provide an efficient support for ANM systems. A biological analogy that can be used is the coherent and stable behavior from cells that form tissues of complex multicellular organisms. Tissues present fault tolerance features in their operation and communication strategies (i.e., cell signaling). In an analogous way, overlay operation could improve service availability replicating messages and operations among different peers.

There are some initiatives investigating P2P-based ANM. In these initiatives, peers have some properties found in Autonomic Management Elements (AME) and peer groups have some properties found in Autonomic Management Domains (AMD). Some of these initiatives do not support every well known features of Autonomic Computing, but, this support is not indispensable for our investigation. For sake of simplicity, only few initiatives in P2P-based ANM will be cited, as follows.

PBMAN [9] merges traditional PBNM with P2P overlays to manage Ambient Networks (AN). PBMAN enables management tasks inside the AN, as well as distribution and retrieval of policies. Through this approach it is possible to establish policies to manage devices or services. PBMAN is structured using super peers, in a hierarchical architecture. Policies are not effectively distributed in this initiative, and, for fault tolerant features, super peers are replicated.

ManP2P [10] is an example of P2P-based network management system that is evolving to an autonomic conception through the implementation of autonomic modules in peers. ManP2P is partially inspired by the Management by Delegation (MbD) model and based on a service-oriented approach. The autonomic modules are designed to support self-* features and to communicate with other components of the ANM system, when necessary. Policies are only used to control overlay parameters in ManP2P and there is no distribution of management decisions [11].

The Madeira platform [5] is an approach to network management topologies that uses the concept of Adaptive Management Components (AMC), which are containers that run on managed elements. Policies, in this approach, are used to dynamically control management parameters. AMCs can manage elements on which they are running and communicate with other AMCs running on other managed elements through P2P communication services. AMCs form management clusters, where a cluster head (i.e., super peer) coordinates the cluster and correlates data. The use of cluster heads brings concerns about scalability and robustness of management clusters.

Despite many improvements brought by P2P-based ANM systems (e.g., higher availability of network management system), there are still issues to be addressed. Policy states in different peers of an P2P-based ANM system can become inconsistent in overlay operation due to faults or lack of synchronization. Besides, the consistency maintenance of policy states must be done maintaining scalability and robustness features of P2P overlays. This consistency maintenance is equally important to any decentralized policy-based ANM system to present a coherent behavior as a whole.

The scenario faced by P2P-based ANM systems to maintain consistency of policy states is similar to scenarios faced in different MAS. Agents must be able to assess and maintain the integrity of the information exchanged among them and conflicts about contradictory knowledge may arise during the communication. Besides, agents have to attend their tasks in an asynchronous way. Thus, techniques from MAS could be used to improve the consistency of policy states in a decentralized ANM system.

III. Proposal

In a decentralized Autonomic Network Management (ANM) system, Autonomic Management Elements (AME) must share management data within an Autonomic Management Domain (AMD). In this work, *management datum* is defined as a management information described in a defined form (i.e., using a specific language). Besides, in these systems, management data must allow their use in automation and/or optimization procedures. It is also expected that sources of management data (e.g., highly dynamic environments) impose challenges to the ANM system. Despite these requirements, it is necessary to avoid potential inconsistencies in state of management data among AMEs.

Our proposal is aimed at meeting consistency requirements of states of management data in a decentralized ANM system. The proposed mechanism introduces *multi-agent truth maintenance* [6] features through a *consistency maintenance module* that runs in each AME. As far as we are aware of, the only studies that incorporate multi-agent truth maintenance features in network management systems were carried out by Nobre and Granville [12] [13]. In this paper, we extend some of our concepts and preliminary results using policy systems [12]. We are aware of only one other work in policy systems that use Truth-Maintenance Systems (TMS) technology, as a mono-agent dependency-tracking mechanism [14].

Multi-agent truth maintenance is a multi-agent extension to TMS. The goal of TMS utilization is to keep the integrity of knowledge bases. In this work, a *knowledge base* is a

group of integrated management data. In a multi-agent TMS, there are multiple agents and each one has its own TMS. TMSs keep integrity performing belief revision and exchange in a set of beliefs. A belief is a member of the current set of beliefs if it has valid reasons. The evaluation of logical consistency of the knowledge base is done by another module, the Problem Solver (PS). The PS exchange beliefs with TMS to maintain the consistency of data states. In policy systems, the PS role is played by the software component that handles policy processing.

The exchange of beliefs about management data is done asynchronously and we do not consider the message exchange to be reliable. Unfortunately, it is well known that the utilization of asynchronous unreliable distributed systems imposes challenges to achieve consistency in shared data [15]. Thus, the consistency model used is non-deterministic, in other words, it uses a "weak" notion of consistency. This model is adopted for scalability, robustness, and update dissemination issues. Given a belief X that depends on some other belief Y, when an update is made to Y, it is eventually reflected in X. Some authors call a similar notion as "eventual consistency" [16].

Belief computations can occur concurrently, thus it is possible that a state (inferred by different AMEs) is based on different set of beliefs. Different participating AMEs in an AMD are not guaranteed to have identical copies of current set of beliefs even if queried at the same time, and not all AMEs are guaranteed to perceive each and every update to the current set of beliefs. To achieve eventual consistency, all belief changes must be replicated among all AMEs within an AMD.

A. Justifications for Management Data

Multi-agent TMS is a kind of *justification-based* TMS. In a justification-based TMS, a datum is believed when it has valid justifications (i.e., valid reasons). This belief can be shared by different AMEs (which play the agent role) in a decentralized ANM system (which aggregates some MAS characteristics). Thus, justifications improve the alignment of AMEs with system-wide objectives (i.e., objectives of the decentralized ANM system). The datum and its list of possible justifications must be provided by network human operators or expert systems for the decentralized ANM system.

For instance, the activation (belief) of a QoS policy (datum) can be justified by a network human administrator command (justification) and an asynchronous signal from a managed device (justification). The code below shows a possible internal representation of this datum and its justifications using ISO *Prolog* standard [17]. In the example, justifications and their presences are represented by facts and the datum is represented by a rule. The "datum(qos_pol)" rule defines that if the justifications "adm_cmd" and "async_sig" are present, the datum "qos_pol" is believed.

```
justification(adm_cmd).
justification(async_sig).
justificationIsPresent(adm_cmd).
justificationIsPresent(async_sig).
datum(qos_pol) :-
```

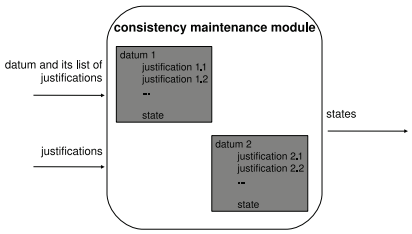

Fig. 1. Consistency maintenance module

```
justificationIsPresent(adm_cmd),
justificationIsPresent(async_sig).
```

The associated states of a datum are "in" (believed) or "out" (disbelieved), according to its justifications. A datum is labeled "out" when it lacks at least one of its associated justifications [6]. We assume that there are not contradictions, so, a datum is labeled "out" only if it has not any of its associated justifications

Justifications can be generated by processes inside the AME or received through the autonomic environment communication services. Thus, the "in" state can assume two additional states: "internal", where the datum has only valid internal justifications, and "external", where the datum has some valid external justification (provided by other AME). In order to enable these additional states, the "justificationIsPresent" fact is replaced by "justificationIsPresent" rules and it is created a new rule, "datumIsInternal". Thus, the presence of a justification is inferred by "generated" or "received" facts. The code below shows the new "justificationIsPresent" version and the "datumIsInternal(qos_pol)" rule.

```
justificationIsPresent(X) :- generated(X).
justificationIsPresent(X) :- received(X).
datumIsInternal(qos_pol) :-
 generated(adm_cmd),
 generated(async_sig).
```

In order to simplify the implementation and analysis issues, we defined that the consistency maintenance module must receive the datum and its list of necessary justifications (structure) as shown in Figure 1. The datum and its list of possible justifications must be provided by network human operators or expert systems. The presence of justifications is based in management information processed by the ANM system. In policy systems, the software component that handles policy processing (i.e., policy module) is in charge of informing changes in justifications to consistency maintenance module.

In this work, we introduce a new possibility for justifications: a datum can be used as a justification for other data. Thus, the presence of this kind of justification is controlled by the state of the datum, in other words, if the datum is "in" it appears as *present* and if the datum is "out" it appears as *absent*. This feature provides support for hierarchical justification and support-list justifications.

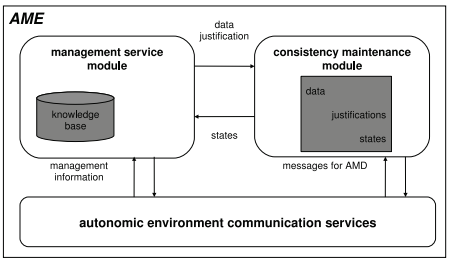

Fig. 2. AME architecture

B. Architecture of Autonomic Management Elements

Autonomic Management Elements (AME) are commonly viewed as composed by one or more *management service modules*. *Management service modules* perform regular management tasks (e.g., collecting statistics) in each AME, and, in these tasks, modules produce management data, building *management knowledge bases*. Furthermore, AMEs can appear and disappear in normal operation, behavior that conventional network management systems do not expect from their constitutive elements.

We introduce the *consistency maintenance module* to register the set of belief about management data in each AME. This module works associating management data and their respective justifications. When there is a belief change (i.e., justification change), the consistency maintenance module uses the autonomic environment communication services to spread the change. Figure 2 shows the relation between the consistency maintenance module, management service modules, and autonomic environment communication services.

Management information is produced in different languages. *A priori*, the consistency maintenance module does not require a particular language for management information. However, the internal representation of management data and justifications must be unique among the AMEs of an AMD. It is important to warn that the responsibility of logical consistency of a datum and its justifications is not in charge of consistency maintenance module. The consistency maintenance module verifies whether the defined justifications for a certain datum are present, and, using this information, controls the state of this datum. An advantage of this approach is that the consistency maintenance module is only triggered when there is a change in belief status of shared data, which means it does not introduce overhead for regular management tasks.

The management service modules should inform the consistency maintenance module about their internal beliefs of management data. The management service modules are also responsible for querying and requiring services from the consistency maintenance module, possibly updating their internal beliefs.

When the presence of a justification is modified, the consistency maintenance module performs the following steps: unlabels the management datum, includes (or removes) the presence of justification and labels the datum again according to new restrictions. The autonomic environment communication services are used to spread changes, which can change beliefs of other AMEs.

The management service modules are responsible for querying and requiring services from the consistency maintenance module, according the demands of their processes. Some similar services employ different strategies such as "*publish/subscribe*" schemes [16] or "*watches* (changes trigger the transmission of a packet) [18]. These strategies could be implemented in the proposed consistency maintenance module, however, initially we keep this operation in charge of management service modules.

It is important to stress that there is only one consistency maintenance module inside an AME, thus, it is not specific to a management service module. Therefore, every management service module in an AME interacts with the same consistency maintenance module. This fact can be explored for the integration of different management services. For instance, a policy processing module, a fault handling module, and a configuration management module (possibly using different languages for representing management data) could be integrated by the consistency maintenance module through justifications.

The consistency maintenance module is offered through a simple interface that management service modules must use. The number of supported operations is restricted to ease the implementation in management service modules. Besides, communication details are not specified in supported operations, so changes in communication strategies do not lead to adaptations in the interfaces of management service modules. Thus, the consistency maintenance module offers an open environment for rapid and dynamic resource integration where federations of management service modules are formed with no central authority.

C. Communication within an Autonomic Management Domain

The consistency maintenance module handles the message exchange through autonomic environment communication services. In this process, requests are adapted in messages to be spread in AMD and vice-versa. The autonomic environment is modeled as an unstructured overlay network, thus there is no relation between the information stored at an AME and its position in the overlay topology.

The methods used for message exchange inside the AMD are modeled using concepts from biology-inspired distributed computing models [7]. Among these models, proliferation-based ones are an interesting choice for communication requirements of our proposal. A belief change possibly undergoes proliferation at AMEs visited, where each AME calculates the probability of forwarding the belief change using a proliferation controlling function. All AMEs in the AMD run exactly the same proliferation controlling function and proliferation can be initiated from any AME in the AMD.

In our previous work [12] [13], we had chosen *replication* (referred in this work as "unbridled replication") as the only

proliferation mechanism in the AMD. In unbridled replication, AMEs spread messages to replicate every change in justifications among the participating entities (i.e., AMEs of a specific AMD). This mechanism is restricted to AMD, fulfilling the criterion of robustness for small peer groups.

In this work, we introduce *controlled replication* as a new proliferation scheme within an AMD. Our goal is to use a much lower number of messages in belief changes (than in unbridled replication). Controlled replication forwarding means some belief changes will not be forward due to restriction rules. The idea behind this control is that this way we can minimize redundant network utilization.

The replication is controlled by a *replication controlling function* (RCF). The RCF defines the probability '*P*' of a belief change to be forwarded to the AMD. Broadly speaking, this probability could be defined using different parameters of the autonomic environment operation. Initially, we define the RCF as shown in Equation 1:

$$P(\rho, \eta b) = \frac{\rho}{1 + \eta b} \tag{1}$$

where ηb represents the number of received messages of a specific belief change within a "backoff delay" (T) and ρ is the positive proliferation constant. The essence of the function is that proliferation of a belief change should decrease with the reception of multiple copies of the same belief change (message). The equation is evaluated at time T.

When the belief change is produced internally, the AME sends belief exchange with $P = 1$ (i.e., the belief change is always sent) to the AMD. When the belief change is received by autonomic environment communication services, the reception of new messages informing the belief change already received during the backoff delay decreases the probability of this belief change to be forwarded to the AMD. Note that when $T = 0$ or $\eta b = 0$, the controlled replication has the same behavior of unbridled replication.

D. Evaluation

The evaluation of our proposal can be performed in different ways. To enable a fully controlled environment for the evaluation, we evaluated the system with simulation. Decentralized ANM is modeled as P2P-based ANM, thus, the AMEs are simulated as peers and peer groups as AMD. In these experiments, we present simulation results that support our scalability and robustness claims. Scalability and robustness are some of the most important motivations for using decentralization in the infrastructure of different systems [19] [7], such as P2P facilities.

As previously stated, we expect that the introduction of multi-agent TMS features keeps desired properties of a decentralized ANM system, maintaining each AME as an independent and self-sustainable entity. As many systems have demonstrated, a system that does not share resources can scale almost infinitely simply by adding constitutive elements (e.g, AMEs in a decentralized ANM system). Besides, maintaining the independence of each AME, single points of failure are eliminated.

The simulation experiments were implemented in Java using *PeerSim* [20], an open source event-based simulator of P2P systems. The system version used has the ability to simulate failures in peers and message exchange, and the overlay is built randomly. The experiments use a simple model of transport layer that can emulate some characteristics, such as loss and delay probabilities. In addition, a peer is chosen randomly as the primary source of changes to not affect measurements and message delay is controlled. Each experiment was conducted at least 10 times.

In the experiments, we use controlled replication as our proliferation scheme. We have presented results from unbridled replication in previous work [12] [13]. Since controlled replication is used, it is necessary to describe the parameters of the Replication Controlling Function (RCF) in each experiment. Besides, all peers in peer groups run exactly the same algorithm. We simulate up to 100,000 peers in a peer group.

In the first experiment ($\rho = 1$, $T = 3$), the number of messages exchanged to spread a justification change in the peer group is measured. This number must be considered as an important cost of the peer group operation, thus, it is important for scalability analysis. Besides, we consider the number of transmitted messages as indicative of network load. In this experiment there were no faults in peers or in message exchange. In Figure 3 we report the measured average and 95% confidence intervals for each peer group size.

Our proposal shows acceptable scalability characteristics on number of exchanged messages within a peer group. The experiment shows that our system behaves as we expected, without stability and convergence problems. The main reason for this is that chosen RCF is cost-effective at covering large peer groups. Even though we do not expect this situation in P2P-based ANM systems and in other decentralized ANM systems, it is important to perform this experiment to infer the behavior trend of peer groups with larger number of participating peers.

In the second experiment, we determined the influence of message loss on the dissemination of a justification change. In this experiment, we varied the message loss probability with following values: 50%, and 75%. During the experiments, we observed a significative correlation between message loss and positive proliferation constant (ρ) on results, thus, we also varied ρ in this experiment with following values: 0.5 and 0.25. In Figure 4, we show the average percentage of coherent (and correct) peers after message exchange to cease.

The experiment shows the influence of message loss and positive proliferation constant (ρ) in the controlled replication scheme. As can be seen from the results in Figure 4, high loss probabilities do lead to less consistency in peer group, but, even with a few participating peers, the percentage of coherent peers is substantial. Besides, more participating peers in peer group and higher values of ρ decrease the influence of loss probability.

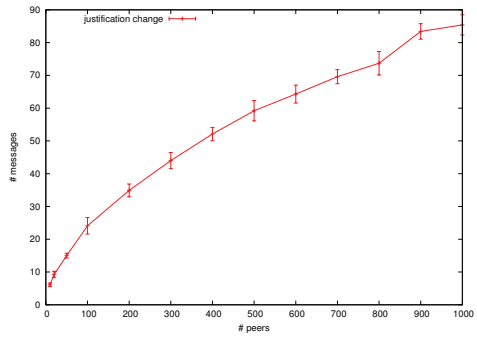

Fig. 3. Message exchange due to a justification change. The error bars indicates the averages and the 95% confidence intervals

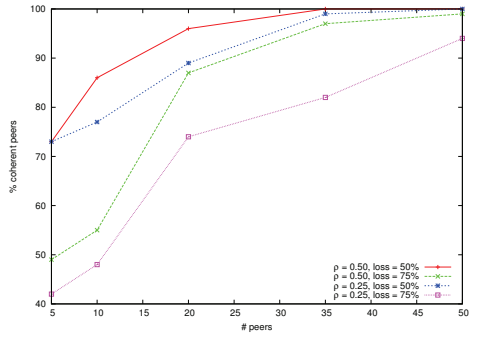

Fig. 4. Coherent peers after a justification change

IV. Case Study

The case study presented is an illustration of the consistency maintenance of policy states in a P2P-based ANM system. We use obligation policies in the form of Event-Condition-Action (ECA) rules as our model. According to our proposal, policy state (i.e., policy activation) is a management datum, and event and conditions are justifications in the consistency maintenance module. The module provides an integration point to enable a coherent global behavior of policy-driven P2P-based ANM systems.

P2P-based ANM can be modeled as a decentralized ANM. In the decentralized model, AME can be organized to form a uniform AMD. In an analogous way, in a P2P-based ANM, peers can be organized into a peer group and then into a P2P overlay. Peers (i.e., AMEs) must be policy-aware to enable an efficient utilization of policies in P2P-based ANM. The consistency maintenance module is responsible to maintain the consistency of the state of management data in the peer group, such as information required from policy processing.

We describe the policy used in this case study using *Ponder2* [21]. Ponder2 is a toolkit that supports the specification and enforcement of policies in the form of ECA rules. When the occurrence of an event is announced to Ponder2, the obligation policy interpreter matches the notification against the registered policy events, hands the event to the relevant policies, which then evaluate their condition(s) and if they succeed, invoke the action(s) specified in the policy.

The code below shows the XML encoding of an obligation policy that will respond to events of type */event/humanResourcesProcedureEvent*. In the example, Ponder2 checks whether the load on the router R1 is low, and the current time is later than 18:00. In this example, if the conditions are satisfied, the policy invokes an action on the router R1 to reserve 10% of bandwidth in "QoS1" profile.

```
<use name="/policy">
 <add name="AdjustQoSPolicy">
  <use name="/template/policy">
   <create type="obligation"
   event= "/event/humanResourcesProcedureEvent"
   active="true">
     <arg name="R1_load"/>
     <arg name="daytime"/>
     <condition>
      <and>
       <eq>!R1_load;<!-- -->low</eq>
       <gt>!daytime;<!-- -->18:00</gt>
      </and>
     </condition>
     <action>
      <use name="/routers/R1">
       <modify profile="QoS1" value="10%"/>
      </use>
     </action>
   </create>
  </use>
 </add>
</use>
```

The consistency maintenance module receives the datum "*AdjustQoSPolicy*" (adj_qos_pol) and its justification list, composed by "*humanResourcesProcedureEvent received*" (hr_proc_evt), "*R1 low load matched*" (R1_load_mat) and "*daytime matched*" (dt_mat). The code below shows the representation of this datum and its justifications. The presence of the justifications is provided by policy processing component or received through the autonomic environment communication services.

```
justification(hr_proc_evt).
justification(R1_load_mat).
justification(dt_mat).
justificationIsPresent(X) :- generated(X).
justificationIsPresent(X) :- received(X).
datum(adj_qos_pol) :-
 justificationIsPresent(hr_proc_evt),
 justificationIsPresent(R1_load_mat),
 justificationIsPresent(dt_mat).
datumIsInternal(adj_qos_pol) :-
 generated(hr_proc_evt),
 generated(R1_load_mat),
 generated(dt_mat).
```

Initially, the datum (i.e., policy state) has the "out" state, since the justifications are not present. This state can be changed after the verification of new beliefs (i.e., justification changes) by consistency maintenance module. These beliefs are informed by police processing component or received as a message in the peer group. If every justification of the datum

is present, the state changes to "in". This "in" state assume two options according the sources of justifications: "internal" (every justification was generated internally) and "external" (some valid external justification).

We will present two situations as examples of use of our proposal: sent events are not received by the policy processing component of one or more peers, and, due to lack of synchronization, some peers disagree about a temporal condition. Both situations could lead to to an inconsistent state of peer group. In our proposal, after receiving an event or changing the evaluation of the temporal condition, policy processing component informs the belief change to the consistency maintenance module. If the new belief implies a justification change, a message will be sent to peer group informing the change. When receiving this message, the consistency maintenance module of other peers checks whether the received justification change is consistent with their knowledge base. If it is not, the consistency maintenance module updates the justification and verifies if this update implies in other changes.

For instance, *human resources procedure event* and *R1 low load matched* can be generated internally and *daytime matched* can be received as a justification change message. In this example, the "in" state assumes the option "external". The code below shows the answer from the consistency maintenance module in this situation.

```
adj_qos_pol:ext (hr_proc_evt:mod R1_load_mat:mod dt_mat:msg)
```

In both cases, even a peer that had not produced some of its justifications, but that had received a justification change message, will be coherent with other peers (i.e., AMEs) of the peer group (i.e., AMD). Thus, this policy state keeps consistent and the P2P-based ANM system show a coordinated behavior. This consistency maintenance of policy states is traditionally performed through a centralized entity, such as super peers. The traditional procedure brings concerns in scalability and robustness and imposes difficulties in the integration of the information.

In this work, we introduce the possibility for using a datum as a justification for another datum. In our example, we can extend the justification "*R1 low load mat*" to include different "loads" of the router R1, such as memory load and cpu load. Thus, we define the datum "*R1 low load mat*" (R1_load_mat) and its justification list, composed by "*cpu low load matched*" (cpu_load_mat) and "*memory low load matched*" (mem_load_mat). The code below shows the representation of this datum and its justifications.

```
justification(cpu_load_mat).
justification(mem_load_ma).
justificationIsPresent(X) :- generated(X).
justificationIsPresent(X) :- received(X).
datum(R1_load_mat) :-
 justificationIsPresent(cpu_load_mat),
 justificationIsPresent(mem_load_mat).
datumIsInternal(R1_load_mat) :-
 generated(cpu_load_mat),
 generated(mem_load_mat).
```

It is required a little change at the representation of the "*AdjustQoSPolicy*" datum. The code below shows the new version of "*AdjustQoSPolicy*" datum.

```
datum(adj_qos_pol) :-
 justificationIsPresent(hr_proc_evt),
 datum(R1_load_mat),
 justificationIsPresent(dt_mat).
datumIsInternal(adj_qos_pol) :-
 generated(hr_proc_evt),
 datumIsInternal(R1_load_mat),
 generated(dt_mat).
```

The feature presented in the last example, the support for hierarchical justification and support-list justifications (as described in Section III), enhances the representation of complex data. Besides, justifications can be use to provide explanations to the user [14], and, using hierarchical justification, we can further improve the understanding of management data [12].

V. Related Work

In this section we first introduce the main concepts behind Truth Maintenance Systems are presented. Finally, some important services related to the maintenance of consistency of shared information in distributed applications are discussed.

A. Truth Maintenance Systems

Truth Maintenance Systems (TMS) were proposed to keep the integrity of Knowledge Bases (KB). The origin of these systems was proposed in the 1970s, for resolutions in mono-agent systems [22]. A TMS keeps track of logical structure of the set of beliefs of agents. A belief is a member of the current set of beliefs if it has valid reasons.

TMSs provide considerable power using few computational resources [14]. Thus, although not being well known outside artificial intelligence community, TMSs are used in different contexts, such as policy systems [14] and network management systems [12] [13].

TMSs have been extended for MAS versions [6]. In a MAS, agents must be able to maintain the integrity of their KBs, despite message exchange with other agents. This maintenance of integrity can be done by a multi-agent TMS. In an analogous way, during the operation of P2P-based ANM system, peers must be able to maintain the integrity of states of management data, despite message exchange with other peers. This similarity indicates the use of multi-agent TMS in P2P-based ANM as an interesting possibility [12] [13].

B. Services for consistency of shared information in distributed systems

Services for consistency of shared information can be used as a basic building block for distributed applications. A decentralized ANM system, as a distributed application, can be improved with characteristics found in these systems.

ZooKeeper [18] is a coordination service for distributed applications. It exposes a simple API that distributed applications can be built upon to implement higher level services for synchronization, data diffusion, and publish-subscribe schemes. *ZooKeeper* use distributed server databases for read operations,

however, write operations use a "leader" server (i.e., centralized database) to assure the consistency of the database.

Astrolabe [16] is a distributed information management service. It works locating and collecting the status of a set of servers and reporting summaries of this information. *Astrolabe* is implemented using a P2P overlay, where every peer run an *Astrolabe* agent (i.e., in a MAS fashion). However, *Astrolabe* was developed primarily using simple data models. Besides, its operation is aimed at read-oriented applications.

The presented efforts show interesting characteristics for consistency of shared information in distributed systems. However, these efforts have vulnerabilities which make them not appropriate for P2P-based ANM systems, such as centralization [18] and simple data models [16].

The consistency of states of management data is still an issue to be addressed in P2P-based ANM systems. It is necessary to maintain this consistency keeping scalability and robustness features of P2P overlays. The maintenance of integrity of KBs among agents brought by multi-agent TMS seems to be a valid direction to introduce this consistency maintenance [12]. Besides, it is interesting to bring characteristics of services for consistency of shared information.

VI. Conclusions and Future Work

The support of new demands faced by traditional network management is a key research issue in the network management area. PBMN has been proving to be a feasible approach for these demands, specially when it is used together with decentralization and automation features. These features can be supported by decentralized ANM systems. An interesting possibility of infrastructure for decentralized ANM system is using P2P overlays. However, the consistency of policy states imposes challenges for these systems.

In this paper we use multi-agent TMS features to improve the consistency of policy states in decentralized ANM systems. Our proposal aims at the integration of data used by the entities that form these systems (i.e., AMEs), through the utilization of justification. We have also presented evaluations of this proposal through simulation experiments. In addition, we have described a case study using obligation policies to show the possibilities of our proposal.

Although the proposal shows good results in evaluations performed until the present moment, it is necessary to evaluate more complicated cases, in number of AMEs and AMDs, and in the participation of an AME in different AMDs. We are also looking at additional settings that could lead to important effects, such as network partitions. Thus, we are currently pursuing new experiments with *PeerSim*.

References

[1] Jennings, B. and Van Der Meer, S. and Balasubramaniam, S. and Botvich, D. and Foghlu, M. O. and Donnelly, W. and Strassner, J., "Towards autonomic management of communications networks," *IEEE Communications Magazine*, vol. 45, no. 10, pp. 112–121, 2007.

[2] Pras, A. and Schoenwaelder, J. and Burgess, M. and Festor, O. and Martinez Perez, G. and Stadler, R. and Stiller, B., "Key Research Challenges in Network Management," *IEEE communications magazine*, vol. 45, no. 10, pp. 104–110, October 2007.

[3] EMANICS, *Activities - Work Package 9 - Autonomic Management*. http://emanics.org/content/view/63/109/, 2009.

[4] Badr, N. and Taleb-Bendiab, A. and Reilly, D. , "Policy-based autonomic control service," in *Policies for Distributed Systems and Networks, 2004. POLICY 2004. IEEE Workshop on*, 2004, pp. 99–102.

[5] Fallon, L. and Parker, D. and Zach, M. and Leitner, M. and Collins, S., "Self-forming Network Management Topologies in the Madeira Management System," *Lecture Notes in Computer Science*, vol. 4543, p. 61, 2007.

[6] Huhns, M. N. and Bridgeland, D. M., "Multiagent truth maintenance," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 21, no. 6, pp. 1437–1445, 1991.

[7] Babaoglu, O. and Canright, G. and Deutsch, A. and Di Caro, G.A. and Ducatelle, F. and Gambardella, L.M. and Ganguly, N. and Jelasity, M. and Montemanni, R. and Montresor, A. and others, "Design patterns from biology for distributed computing," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 1, no. 1, pp. 26–66, 2006.

[8] Granville, L. Z. and da Rosa, D. M. and Panisson, A. and Melchiors, C. and Almeida, M. J. B. and Tarouco, L. M. R. , "Managing computer networks using peer-to-peer technologies," *IEEE Communications Magazine*, vol. 43, no. 10, pp. 62–68, 2005.

[9] Kamienski, C. and Fidalgo, J. and Sadok, D. and Lima, J. and Pereira, L. and Ohlman, B., "PBMAN: A Policy-based Management Framework for Ambient Networks," in *Policies for Distributed Systems and Networks, 2006. POLICY 2006. IEEE Workshop on*, 2006, pp. 79–83.

[10] Panisson, Andre and da Rosa, Diego M. and Melchiors, Cristina and Granville, Lisandro Z. and Maria and Liane, "Designing the Architecture of P2P-Based Network Management Systems," in *ISCC '06: Proceedings of the 11th IEEE Symposium on Computers and Communications*. IEEE Computer Society, 2006, pp. 69–75.

[11] Marquezan, C. C. and dos Santos, C. R. P. and Nobre, J. C. and Almeida, M. J. B. and Tarouco, L. M. R. and Granville, L. Z., "Self-managed Services over a P2P-based Network Management Overlay," in *Proceedings. 2nd Latin American Autonomic Computing Symposium (LAACS 2007)*, 2007.

[12] Nobre, J. C. and Granville, L. Z., "Towards Consistency of Policy States in Decentralized Autonomic Network Management," *Policies for Distributed Systems and Networks, IEEE International Workshop on*, vol. 0, pp. 170–173, 2009.

[13] ——, "Consistency of States of Management Data in P2P-Based Autonomic Network Management," in *20th Ifip/Ieee International Workshop on Distributed Systems: Operations and Management, DSOM 2009, Venice, Italy, October 27-28, 2009, Proceedings*. Springer-Verlag New York Inc, 2009, p. 99.

[14] Kagal, L. and Hanson, C. and Weitzner, D., "Using Dependency Tracking to Provide Explanations for Policy Management," in *Policies for Distributed Systems and Networks, 2008. POLICY 2008. IEEE Workshop on*, 2008, pp. 54–61.

[15] Fischer, M.J. and Lynch, N.A. and Paterson, M.S., "Impossibility of distributed consensus with one faulty process," *Journal of the ACM (JACM)*, vol. 32, no. 2, pp. 374–382, 1985.

[16] Van Renesse, R. and Birman, K.P. and Vogels, W., "Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining," *ACM Transactions on Computer Systems (TOCS)*, vol. 21, no. 2, pp. 164–206, 2003.

[17] Scowen, R., "Technical Report ISO/IEC DIS 13211-1: 1995 (E)," International Organisation for Standardisation, Geneva, Switzerland, Tech. Rep., 1995.

[18] Hunt, P., *ZooKeeper: A Distributed Coordination Service for Distributed Applications*. http://wiki.apache.org/hadoop/ZooKeeper, 2008.

[19] J. A. Mccann and M. C. Huebscher, "Evaluation Issues in Autonomic Computing," in *Grid and Cooperative Computing – GCC 2004*, ser. Lecture Notes in Computer Science, vol. 3252. Heidelberg, Springer-Berlim, 2004, pp. 597–608.

[20] Jelasity, M. and Montresor, A. and Jesi, G. and Voulgaris, S., *The Peersim Simulator*. http://peersim.sf.net, 2008.

[21] Ponder2 Documentation, "http://www.ponder2.net/."

[22] Jon Doyle, "A truth maintenance system," *Computation & intelligence: collected readings*, pp. 529–554, 1979.