Dissertação de Mestrado

Inatel
Instituto Nacional de Telecomunicações

DESACOPLAMENTO DE IDENTIFICADORES E LOCALIZADORES: UMA COMPARAÇÃO DE ABORDAGENS E PROPOSTA DE ARQUITETURA

BRUNO MAGALHÃES MARTINS

FEVEREIRO 2011

Desacoplamento de identificadores e localizadores: uma comparação de abordagens e proposta de arquitetura

BRUNO MAGALHÃES MARTINS

Dissertação apresentada ao Instituto Nacional de Telecomunicações, como parte dos requisitos para obtenção do Título de Mestre em Telecomunicações.

Orientador: PROF. DR. ANTÔNIO MARCOS ALBERTI

Santa Rita do Sapucaí
2011

Dissertação defendida e aprovada em 18/02/2011, pela comissão julgadora:

_____

Prof. Dr. Antônio Marcos Alberti (Orientador) - Inatel

_____

Prof. Dr. José Marcos Camara Brito - Inatel

_____

Prof. Dr. Fábio Luciano Verdi - Universidade Federal de São Carlos

_____

Prof. Dr. Luciano Leonel Mendes - Coordenador do Curso de Mestrado

Dedico este trabalho a meus pais, irmãos e a minha noiva.

"*Bem-aventurado o homem que acha sabedoria, e o homem que adquire conhecimento; porque é melhor a sua mercadoria do que artigos de prata, e maior o seu lucro que o ouro mais fino*". Provérbios 3:13,14.

Agradecimentos

Agradeço primeiramente a Deus e Nossa Senhora por iluminar-me, concedendo saúde e paz para conseguir vencer mais uma etapa da minha vida acadêmica. Aos meus pais e irmãos por me compreenderem, me apoiarem, pelo incentivo e força espiritual para conseguir transpor os obstáculos enfrentados durante o curso de mestrado. A minha noiva, Maria Tereza, por me acompanhar nos momentos difíceis, pela paciência, compreensão e pelas palavras de incentivo;

Ao Instituto Nacional de Telecomunicações por me conceder a oportunidade de fazer o curso de mestrado. A todos os professores do curso de mestrado do INATEL, especialmente ao meu orientador Dr. Antônio Marcos Alberti por dividir comigo o seu vasto conhecimento, pela oportunidade, atenção e paciência no processo de orientação;

Aos meus amigos e colegas de mestrado pela ajuda e companheirismo prestados durante estes dois anos.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

4WARD	*Architecture and Design for the Future Internet*
ACK	*Acknowledgment*
AH	*Authentication Header*
AId	*Application Identifiers*
API	*Application Programming Interface*
AR	*Attachment Register*
ASCII	*American Standard Code for Information Interchange*
BGP	*Border Gateway Protocol*
BO	*Binary Object*
CCN	*Content-centric Network*
CH	*Correspondent Host*
CN	*Correspondent Node*
CoA	*Care-of Address*
DFZ	*Default-free Zone*
DHT	*Distributed Hash Table*
DLS	*Distributed Location Services*
DNS	*Domain Name System*
DO	*Data Object*
DoS	*Denial-of-Service*
ESP	*Encapsulating Security Payload*
ETR	*Egress Tunnel Router*
FIB	*Forwarding Information Base*

FIds	*Forwarding Identifiers*
FN	*Forwarding Nodes*
FP7	*European Union Framework Programme 7*
GIRS	*Generalized Indirection Resolution System*
GPS	*Global Positioning System*
HI	*Host Identifier*
HIP	*Host Identity Protocol*
HMS	*HUI Mapping Sublayer*
HUI	*Hierarchical URI-like Identifier*
IETF	*Internet Engineering Task Force*
IMS	*Identity Management Server*
IO	*Information Object*
IP	*Internet Protocol*
IPSec	*Internet Protocol Security*
IPv4	*Internet Protocol version 4*
IPv6	*Internet Protocol version 6*
IRTF	*Internet Research Task Force*
ISP	*Internet Service Provider*
ITF	*Inter-domain Topology Formation*
ITR	*Ingress Tunnel Router*
IWT	*International Workshop on Telecommunications*
JPEG	*Joint Photographic Experts Group*
KISS	*Keep It Simple, Stupid*
LISP	*Locator ID Separation Protocol*
LLC	*Late Locator Construction*
LS	*Location Services*
MAC	*Media Access Control*
MB	*Mecanismo de Busca*
MILSA	*Mobility and Multihoming Supporting Identifier Locator Split Architecture*
MN	*Mobile Node*

MP3	*MPEG 1 Layer-3*
MPEG	*Moving Picture Experts Group*
NAI	*Network Access Identifier*
NetInf	*Network of Information*
NRS	*Names Resolution System*
OSPF	*Open Shortest Path First*
OWL	*Ontology Web Language*
P2P	*Peer-to-Peer*
PARC	*Palo Alto Research Center Incorporated*
PIT	*Pending Interest Table*
PKI	*Public Key Infrastructure*
PLA	*Packet Level Authentication*
PSIRP	*Publish/Subscribe Internetworking Routing Paradigm*
RFC	*Request for Comments*
RFID	*Radio Frequency Identification*
RIds	*Rendezvous Identifiers*
RLOCs	*Routing Locators*
ROFL	*Routing on Flat Label*
RPC	*Remote Procedure Call*
RTFM	*Rendezvous, Topology, Forwarding and Mediation*
RZBS	*Realm Zone Bridging Server*
SCTP	*Stream Control Transmission Protocol*
SIds	*Scope Identifiers*
TCP	*Transport Control Protocol*
TM	*Topology Management*
TTL	*Time to Live*
UDP	*User Datagram Protocol*
UE	*end-User Equipment*
URI	*Uniform Resource Identifier*
URL	*Uniform Resource Locator*
XML	*Extensible Markup Language*

Resumo

Neste trabalho são apresentadas e comparadas algumas abordagens que tratam o desacoplamento de identificadores e localizadores de *hosts* e de conteúdos em redes atuais e futuras. Tais abordagens propõem soluções para diversos problemas como o de mobilidade e *multihoming* de entidades físicas e lógicas, bem como para o problema de escalabilidade, segurança e de distribuição de conteúdos. Uma vez levantadas as principais vantagens de cada abordagem, é proposto e especificado um Sistema Generalizado de Resolução de Indireções com o objetivo de identificar unicamente as entidades, bem como nomeá-las e descrevê-las com base em informações legíveis, além de mapear de forma dinâmica a identificação na localização destas entidades.

Palavras-chave: Desacoplamento de identificadores e localizadores, indireções, redes centradas no conteúdo, tabelas *hash* distribuídas.

Abstract

This work presents and compares some approaches that deal witth the hosts and content split identifier/locator in current and future networks. Such approaches propose solutions to several problems such as mobility and multihoming of physical and logical entities, as well as the problem of scalability, security and content distribution. Once the main advantages of each approach have been highlighted, we propose a Generalized Indirections Resolution System in order to uniquely identify the entities, as well as name and describe them based on legible information besides dynamically mapping the attributes (identifier, name, descriptor and locator) of these entities.

keywords: Split ID-Loc, indirections, content-centric networks, distributed hash tables.

Capítulo 1

Introdução

1.1 Contextualização e Motivação

A Internet é apoiada em princípios elaborados há mais de 40 anos, quando recursos de memória, processamento e comunicação eram muito limitados. Seu tremendo sucesso e a diversidade de aplicações originadas têm feito exigências para muito além das quais ela foi originalmente proposta. Tal uso indiscriminado tem ocasionado diversos problemas, dentre eles, problemas de escalabilidade, suporte à mobilidade, *multicast*, *multihoming*, distribuição de conteúdos, identificação única e localização de entidades (HARAI, et al., 2007). De forma geral, a solução destes problemas tem sido criar incrementos/remendos à arquitetura. Entretanto, esta abordagem tem criado uma verdadeira "colcha de retalhos", o que dificulta a evolução da rede, impedindo soluções mais significativas para os problemas existentes.

Esta abordagem incremental apóia-se sobre uma visão evolutiva da arquitetura IP (*Internet Protocol*). Em tal visão, as propostas de melhorias tomam-se soluções pontuais (*patches*), tratando cada um dos problemas de forma individual, ao invés de uma solução unificada para todos os problemas das redes IP atuais. Um exemplo de abordagem incremental são os protocolos de desacoplamento da identificação e localização de *hosts*. Estes protocolos visam uma solução incremental para os problemas de mobilidade e de *multihoming* do protocolo IP. Entretanto, outras arquiteturas com abordagens unificadas, projetadas a partir do zero e que não são obrigatoriamente atreladas às redes atuais têm sido propostas. Estas arquiteturas são baseadas nas abordagens *Clean Slate*.

De acordo com a Universidade de Stanford em (STANFORD UNIVERSITY, 2010), o termo *Clean Slate* significa reiniciar do zero com o que já se sabe hoje, o desenvolvimento de arquiteturas, protocolos e sistemas de comunicação visando resolver todos os problemas atuais e construir soluções mais inteligentes, adaptativas e que fazem o melhor uso dos recursos como os de transmissão, processamento, energia e memória sem ter como limitante a manutenção da compatibilidade com os sistemas atuais.

Ainda sobre *Clean Slate*, para Bellovin (BELLOVIN et al., 2005), tal abordagem foca os projetos para uma direção que não sejam vinculados aos requisitos impostos pelas redes atuais. O desafio não é adaptar estas redes para suprir as novas demandas, mas fazer uma especulação das necessidades futuras, trabalhar cuidadosamente nas exigências arquitetônicas, instigar a pesquisa em conceitos inovadores de redes para atender tais requisitos, além disso, deve-se produzir um argumento concreto a favor da arquitetura proposta.

As redes centradas na informação são um exemplo da abordagem *Clean Slate* (JACOBSON, et al., 2009). Elas propõem uma solução unificada para boa parte dos problemas das redes IP. A ideia é criar uma rede de abrangência mundial que direciona todo foco na informação propriamente dita. Estas redes procuram facilitar a localização e a disseminação da informação através do uso de eficientes mecanismos de pesquisa e roteamento destas informações. Estas redes se beneficiam do armazenamento da informação em *caches* e de novos paradigmas de roteamento baseados em nomes.

Entretanto, esta nova abordagem sofre com o problema da identificação e nomeação das entidades, uma vez que nas redes centradas na informação, cada objeto de informação deve ter um identificador único para auxiliar a localização, a disseminação desta informação e ao mesmo tempo evitar ambigüidades e redundâncias desnecessárias. Em outras palavras, o identificador de um objeto de informação deve ter o mesmo identificador de uma cópia deste objeto armazenado em um local diferente, ou seja, a localização não deve influenciar semanticamente na construção do identificador deste objeto.

Com isso, forma-se um desacoplamento lógico entre a identificação e a localização destes objetos de informação. Entretanto, para a recuperação desta informação é necessário um sistema que faça o mapeamento dinâmico entre os identificadores e os localizadores tanto dos objetos de informação em questão, quanto dos *hosts* responsáveis pelo armazenamento destes objetos.

1.1.1 Desacoplamento da identificação e localização em *hosts*

A crescente proliferação de dispositivos móveis com acesso à Internet tem tornado cada vez mais evidentes algumas das limitações desta rede. A arquitetura TCP/IP (*Transport Control Protocol / Internet Protocol*) sobrecarrega o endereço IP com as funcionalidades de identificação e localização de *hosts*. Ou seja, o endereço IP tem dupla funcionalidade e mantém um acoplamento lógico entre o identificador e o localizador dos *hosts*. Com isso, as conexões se perdem quando o endereço IP é modificado para atualizar a localização de um nó móvel, bem como para atualizar um evento de *renumbering*. Esta dupla funcionalidade causa diversos problemas na Internet atual, principalmente no suporte à mobilidade, na comunicação *multicast* e ao realizar *multihoming*.

A garantia de mobilidade é um dos principais desafios ao projetar uma rede de nova geração, ou seja, a garantia de que os usuários podem se mover não apenas dentro da sua rede local, mas também mudar de rede de acesso sem que a conectividade seja interrompida. Além do acoplamento lógico entre identificadores e localizadores de *hosts*, o suporte à mobilidade encontra desafios na localização do dispositivo móvel, no roteamento de pacotes para estes dispositivos, na sinalização da mudança da rede residente para a visitada e no rastreamento autorizado de usuários e terminais em caso de comportamento ilícito (IN, et al., 2007).

O termo *multihoming* significa que existem múltiplas possibilidades de conexões para o acesso à Internet através de uma rede ou de uma estação. Ou seja, significa ter múltiplas interfaces com múltiplos localizadores para um mesmo *host*, ao mesmo tempo. As principais funções do *multihoming* são oferecer redundância e otimização de rede, além de dar suporte

para selecionar o ISP (*Internet Service Provider*) que ofereça melhor recurso para um determinado serviço.

O grande desafio encontrado em comunicações *multicast* é enviar um fluxo de informação para um conjunto de destinatários de forma eficiente. A eficiência se dá evitando redundância desnecessária, ao enviar um único fluxo da mesma informação através de um meio de transmissão e compartilhá-lo apenas quando os destinatários estiverem ligados por enlaces diferentes.

Com o desacoplamento da identificação e localização dos *hosts* na rede, os identificadores são utilizados nas camadas de aplicação e transporte para fim de identificação do *host*, enquanto os localizadores são utilizados na camada de rede para localizar o *host* logicamente na topologia desta rede. Com base neste princípio, soluções para o problema da utilização de um único endereço para identificação e localização de *hosts* têm sido propostas. Algumas delas serão listadas e analisadas qualitativamente no Capítulo 2.

1.1.2 Redes Centradas na Informação

Diante da disseminação do uso da Internet e do aumento da acessibilidade aos dispositivos de criação e modificação de mídia, hoje as redes passam por uma fase em que grande parte dos usuários conectados está deixando de ser mero consumidor, para se tornar produtor de conteúdos. A maioria destes conteúdos (fotos, vídeos, *posts*, etc.) é disponibilizada na rede e o grande problema é que na arquitetura atual toda a preocupação é centrada nos *hosts*, não nos conteúdos gerados e armazenados por estes *hosts*. Em outras palavras, os usuários estão interessados no conteúdo que a rede disponibiliza, enquanto a Internet atual é focada na localização dos *hosts* que armazenam este conteúdo. Há uma série de problemas oriundos desta incompatibilidade de modelos, tais como problemas na disponibilidade do conteúdo, de segurança e de dependência da localização do conteúdo. Para

Van Jacobson[1] (JACOBSON, et al., 2009) a forma direta e unificada de resolver estes problemas é substituir o "onde" pelo "o que" em relação ao conteúdo.

Esta incompatibilidade se deve ao fato de que a Internet atual foi projetada utilizando o paradigma *host-centric*, que colocou os terminais da rede no centro do projeto original. Como resultado, a rede evoluiu para conectar *hosts* através do roteamento IP. Funções complexas foram retiradas do núcleo da rede e movidas para os terminais. Levantamentos recentes (JACOBSON, et al., 2009) sugerem que parte significativa do tráfego atual da Internet é voltada a transferência de conteúdos, e não mais para as aplicações originais da rede. Novos modelos de comunicação, dentre eles o *peer-to-peer*, surgiram para melhorar a distribuição e a troca de conteúdos na Internet. Entretanto, nos últimos anos o modelo *host-centric* começou a ser questionado e um novo paradigma apareceu: o *information-centric* (ou redes centradas na informação). Já que a Internet hoje basicamente é uma rede de transferência de conteúdos e informações, porque não centrar sua evolução neste aspecto ao invés de centrar a sua evolução nos terminais.

Diversas linhas de pesquisas veem a mudança para o foco no conteúdo como a transição para a terceira geração de redes, sendo que a primeira geração foi caracterizada por ambientes totalmente com fio e a disseminação da telefonia; a segunda é sobre interconexões com fio baseadas na pilha de protocolos TCP/IP; e a terceira geração de redes seriam as redes centradas no conteúdo, que são caracterizadas pela interconexão e disseminação da informação em larga escala e pelo suporte à mobilidade. Neste paradigma, todas as ações da rede são focadas no processamento e troca de informação, independentemente de onde esta informação esteja localizada (ESTEVE, et al., 2008).

1 Van Jacobson é um dos principais contribuintes para a base tecnológica da Internet de hoje. Ele é o autor original do cabeçalho TCP/IP e trabalha atualmente como pesquisador do PARC (*Palo Alto Research Center Incorporated*) no programa de investigação de redes centradas em conteúdo. Van Jacobson afirma que a CCN (*Content-centric Network*) preservará as decisões de projeto do TCP/IP como a simplicidade, a robustez e a escalabilidade (JACOBSON, et al., 2009) (PALO ALTO RESEARCH CENTER INCORPORATED, 2010).

Nas redes centradas na informação, as informações são dirigidas para os nós que manifestam o seu interesse através de assinaturas de conteúdos, em vez de nomes de interfaces de *hosts* (ESTEVE, et al., 2008). Do ponto de vista do usuário, toda informação terá um identificador único e esta informação poderá ser recuperada de forma mais eficiente, já que a informação pode ser encontrada em um local mais próximo ou até mesmo no seu ambiente local, além do aumento do desempenho e a facilidade no acesso a estas informações.

As redes centradas na informação possuem como funcionalidade principal a interconexão dos produtores de conteúdo aos consumidores, independente das localizações dos *hosts* envolvidos na comunicação. Desta forma, consegue-se um aumento da eficiência na disponibilidade e disseminação da informação, já que a recuperação dos conteúdos pode ser beneficiada ao utilizar cópias ao longo da rede.

O grande desafio das redes centradas na informação é, portanto, estabelecer um padrão para nomeação da informação de forma consistente e eficiente, além de fornecer soluções para roteamento das informações baseado em nomes. No Capítulo 2 são discutidas algumas propostas de redes centradas na informação, apresentado suas características e comparado-as.

1.2 Proposta e contribuições do trabalho

O objetivo desta dissertação é a análise e comparação de arquiteturas de redes atuais e futuras que tratam os problemas envolvidos no processo de identificação e nomeação tanto de dispositivos quanto de conteúdos. Com base nos requisitos e desafios levantados na fase de análise, é proposto e especificado um Sistema Generalizado de Resolução de Indireções para tratar o problema da identificação única, nomeação e localização de entidades reais e virtuais em rede.

Durante a execução deste trabalho foi submetido e aceito o seguinte artigo para o IWT 2011 (*International Workshop on Telecommunications*):

- MARTINS, Bruno M., ALBERTI, Antônio M.: **Desacoplamento de Identificadores e Localizadores em Redes Atuais e Futuras: Uma Comparação de Abordagens e Proposta de Arquitetura.** 2010.

1.3 Organização da dissertação

O restante desta dissertação é organizado como segue: O Capítulo 2 contém um estudo da estrutura e das características de um sistema de nomeação e o levantamento dos desafios encontrados para a gerência deste tipo de sistema. Além disso, é feito um estudo e comparação de algumas propostas de arquiteturas que tratam os desafios da identificação e localização de entidades em rede, incluindo propostas de desacoplamento de identificadores e localizadores e de propostas de redes centradas na informação. No Capítulo 3 é abordado o conceito de indireções e a relação entre indireções e as propostas de arquiteturas estudadas no Capítulo 2. Ainda no Capítulo 3 é explicado e exemplificado o funcionamento das DHTs (*Distributed Hash Tables*) com o objetivo de tratar indireções a serem propostas no Capítulo 4. O Capítulo 4 contém a estrutura e o funcionamento da proposta principal deste trabalho, um Sistema Generalizado de Resolução de Indireções, além de um estudo de caso para a utilização destas indireções em um contexto de publicação e assinatura de objetos de conteúdo; e, finalmente, o Capítulo 5 contém a conclusão desta dissertação, bem como algumas propostas de trabalhos futuros.

Capítulo 2

Desacoplamento de identificadores e localizadores

2.1 Sistemas de nomeação

Genericamente, os nomes são utilizados para identificar entidades como pessoas, dispositivos de rede, conteúdos, dentre outras entidades que se relacionam em um contexto global. Desta forma, é interessante que os nomes sejam criados com significado contextual, sempre que possível, já que estes nomes serão utilizados como uma espécie de rótulo para as entidades nomeadas.

Os nomes podem ser opacos, ou seja, não interpretáveis por humanos. Por outro lado, nomes legíveis possuem significado para as pessoas que os utilizam. Os nomes legíveis são construídos com base na semântica e/ou no contexto da entidade nomeada. Para exemplificar, dispositivos nomeados de forma legível podem ser reconhecidos e localizados através do seu significado, ou em outras palavras, um nome legível de um dispositivo pode descrevê-lo ou descrever a sua função dentro de um determinado contexto, por exemplo, *impressora01 ou computador_vendas*.

Entretanto, nem sempre é possível criar nomes legíveis, dado que alguns sistemas de nomes utilizam propriedades que "mascaram" os nomes ou não utilizam características semânticas para nomear as entidades. Um exemplo de nomes opacos são os criados através de chaves criptográficas que visam melhorar a segurança, bem como os nomes que são gerados utilizando funções *hash*.

Uma função *hash* é composta usualmente por uma função de compressão de dados que recebe um valor de tamanho arbitrário e retorna um código comprimido de tamanho fixo para este valor. Esta função de compressão gera um resultado "resistente a colisões", ou seja, é estatisticamente impossível dois valores diferentes entre si gerar um mesmo resultado *hash*. Apesar das funções *hash* criarem nomes opacos, elas formam um conceito eficiente para serem utilizadas em pesquisas de valores a partir de uma dada chave (BIHAM & DUNKELMAN, 2009) e (STEVENS, 2007).

Um sistema de nomes geralmente é constituído por: (i) um espaço de nomes (*namespace*), definido por este sistema de nomes a fim de delimitar os aspectos nos quais os nomes deverão ser constituídos. Em outras palavras, o *namespace* define as regras para a estrutura, a utilização e a validação dos nomes; (ii) um registro de nomes que é responsável pelo processo de vinculação de nomes a entidades específicas, como por exemplo, um nome a um endereço; (iii) e um processo de resolução de nomes responsável por mapear dinamicamente um nome (ou um identificador) para um determinado endereço (KOZIEROK, 2005).

Um dos principais exemplos de sistemas de nomes é o DNS (*Domain Name System*) especificado pela RFC-1034 (*Request for Comments*) (MOCKAPETRIS, 1987). O DNS é responsável pelo mapeamento de um identificador baseado em nomes para um identificador (ou localizador) baseado em endereço IP. Este tipo de mecanismo permite que os computadores usem simples e eficientes endereços numéricos, enquanto que os usuários possam usar localizadores baseados em nomes, sendo mais fácil de lembrar e mais "amigáveis ao usuário".

A **Figura 1** ilustra um cenário no qual um método de resolução de nomes é usado para determinar qual é o endereço associado a um determinado nome. Antes desta resolução, é necessário, portanto, a atribuição ou vinculação de um endereço ao nome. Esta vinculação é solicitada pelo usuário (com tal permissão) ao registro de nomes. Tanto a interação do *namespace* com o registro de nomes quanto a do *namespace* com a resolução de nomes são classificadas como funções descritivas, uma vez que estas interações fornecem os métodos específicos descritos pelo *namespace* para o registro e a resolução de nomes, respectivamente.

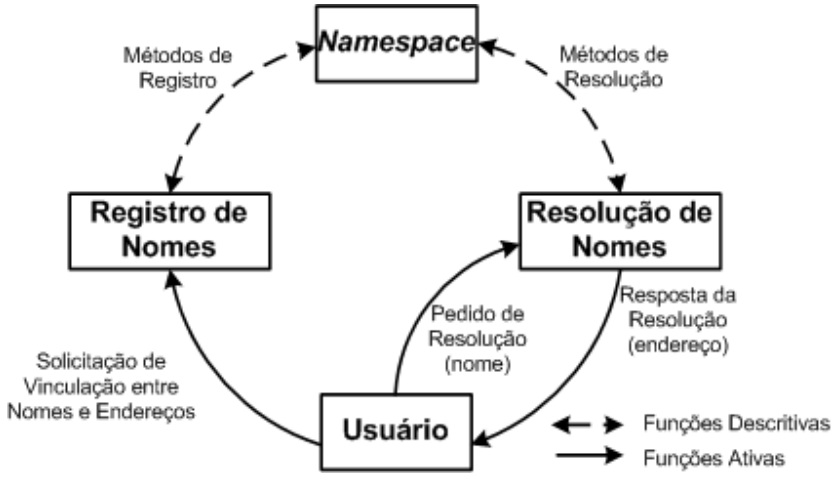

Figura 1. *Sistema de nomeação (KOZIEROK, 2005).*

Como visto, o *namespace* contém as principais características de um sistema de nomes, incluindo a forma como os nomes são registrados e usados, além da relação que estes nomes possuem com os demais nomes e com a própria entidade de gerenciamento. Por esta razão, o *namespace* muitas vezes é denominado de arquitetura de nomes.

Resumidamente, uma arquitetura de nomes provê uma forma para identificar e gerenciar entidades, como pessoas, dispositivos e objetos de informação. Estas arquiteturas podem ser concebidas de forma plana ou hierárquica. Plana no sentido de sem hierarquia. Os nomes planos não possuem qualquer hierarquia que os organiza ou que determina a forma como eles devem ser concebidos, além de não delegarem nem dividirem a responsabilidade do gerenciamento das entidades nomeadas em regiões administrativas. A **Figura 2** ilustra a forma como os nomes planos são concebidos em um contexto de dispositivos interligados em uma organização.

Figura 2. *Arquitetura de nomes planos.*

As arquiteturas de nomes hierárquicos, por sua vez, criam nomes levando em conta a estrutura organizacional nas quais estes nomes estão inseridos. Em outras palavras, os nomes hierárquicos são concebidos em uma estrutura com semântica do tipo "pai/filho". A **Figura 3** ilustra a utilização de nomes formados hierarquicamente.

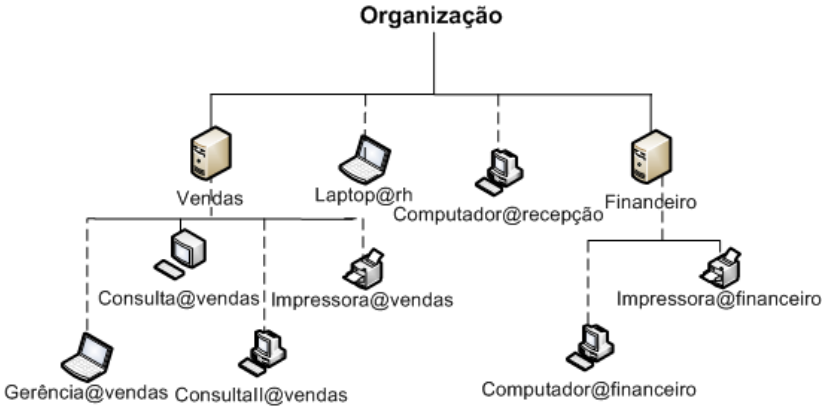

Figura 3. *Arquitetura de nomes hierárquicos.*

2.1.1 Nomeação da informação

Em qualquer projeto de redes centradas na informação deve ser possível se referir à própria informação, independentemente das redes em que ela se encontra. Um esquema de nomeação para objetos de informação é, portanto, a parte mais importante do *projeto* destas arquiteturas (AHLGREN, et al., 2010). Contudo os sistemas de nomes para as redes centradas na informação devem ser criados considerando que:

- o esquema de nomeação deve ser genérico, possibilitando nomear qualquer tipo de entidade, com propriedades e atributos diferentes, incluindo objetos estáticos e dinâmicos, nós de rede, pessoas, entidades do mundo real tais como lugares e objetos;

- o esquema de nomeação deve ser projetado para ser flexível e expansível, permitindo integrar outras propriedades, como as de segurança, para controle de acesso, por exemplo;

- o esquema de nomeação deve fornecer uma maneira global e inequívoca para identificar as informações;

- a cada entidade diferente deve ser associado um identificador (ID) único;

- a segurança deve ser relacionada diretamente à informação, e pode ser garantida através do esquema escolhido para nomeá-la. A cada objeto de informação deve ser dado um ID único com propriedades de segurança nativas do objeto de informação;

- o esquema de nomeação deve possuir um mecanismo de resolução de nomes que seja escalável.

2.2 Desacoplamento da identificação e localização de *hosts*

Assim como citado anteriormente, nas redes atuais o endereço IP é sobrecarregado ao se responsabilizar pela dupla funcionalidade de identificação e localização dos *hosts,* ou seja, a mobilidade na rede muitas vezes leva a troca do endereço do *host*, o que pode ocasionar indiretamente a troca da sua identificação. É como se uma pessoa trocasse de identificação ao mudar de endereço, o que não faz sentido. Esta é uma das causas da dificuldade de identificar o originador de um ataque em uma rede IP. Entretanto, as propostas de desacoplamento entre o identificador e o localizador de *hosts* permitem a melhoria do suporte à segurança, mobilidade e do *multihoming* destes *hosts*.

Existem diversos protocolos e projetos de desacoplamento de identificadores e localizadores de *hosts*. A grande maioria baseada em IP. Os protocolos *Mobile* IP, HIP (*Host Identity Protocol*), LISP *(Locator ID Separation Protocol*) e MILSA (*Mobility and Multihoming Supporting Identifier Locator Split Architecture*) são algumas das propostas que freqüentemente aparecem na literatura. Nas próximas seções estas propostas serão analisadas qualitativamente determinando oportunidades e desafios para pesquisas futuras.

2.2.1 *Mobile* IP

O *Mobile* IP (RFC-3344) (PERKINS, 2002) foi projetado pelo IETF (*Internet Engineering Task Force*) como uma proposta para garantir mobilidade aos dispositivos da rede, designando dois endereços IP para um mesmo dispositivo. O *home-address*, endereço estático que pode ser usado, por exemplo, como um identificador do nó na camada de aplicação e o *care-of-address*, que indica a localização lógica do nó na rede. Este último é associado dinamicamente à posição atual do dispositivo na rede em que se encontra.

Para viabilizar esta solução, o *Mobile* IP necessita de dois componentes na sua arquitetura: o agente local e o agente estrangeiro, responsáveis por atribuir ao *host* o *home-address* e o *care-of-address*, respectivamente. Um nó móvel recebe anúncios periódicos e deduz quando ele mudou de rede por deixar de receber anúncios do agente local e passar a receber anúncios do agente estrangeiro (RAMACHANDRAN, 2005).

Os dados enviados ao terminal móvel (MN – *Mobile Node*) são interceptados pelo agente local, que fica responsável por armazenar a posição atual do nó móvel. Este por sua vez, encapsula os dados e os retransmite ao agente estrangeiro relativo à rede visitada, que finalmente, retransmite os dados ao nó móvel. Para este fim, faz-se necessário manter um mapeamento (ou indireção) do *home-address* com o seu respectivo *care-of-address*. Desta forma, a utilização do *Mobile* IP propõe que os dispositivos móveis possam mudar de rede sem que exista a perda da sua conectividade à rede. A **Figura 4** ilustra o funcionamento do *Mobile* IP.

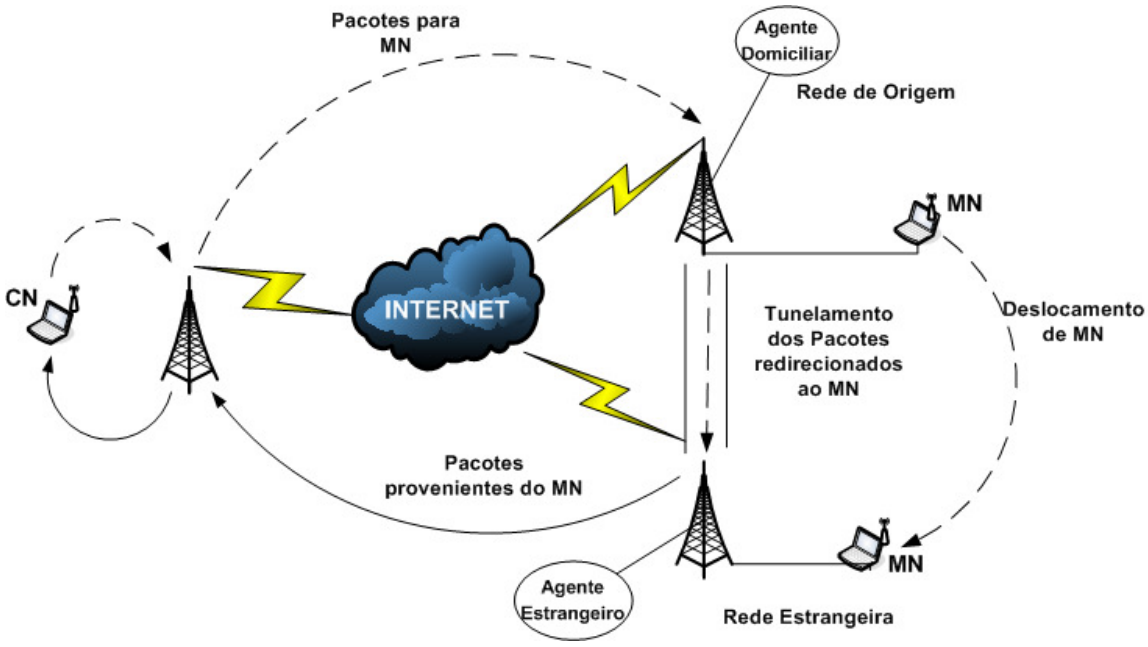

Figura 4. *Funcionamento do Mobile IP (BALDO, 2007).*

Apesar do suporte à mobilidade oferecida pelo *Mobile* IP e da grande utilização da sua ideia original em redes de celulares, em (RAMACHANDRAN, 2005) mostra-se que há desvantagem na eficiência da comunicação, já que existe o acréscimo de um certo número de *bytes* por cabeçalho devido ao tunelamento, além do problema do roteamento triangular, no qual um pacote destinado a um dispositivo móvel precisa visitar a rede de origem antes de ser enviado para a sua localização atual.

O *Mobile* IP com suporte a IPv6 (JOHNSON, et al., 2004) tem os seus princípios semelhantes ao IPv4, o qual também permite que um nó se mova de uma rede a outra sem a perda de conectividade. Porém, o *Mobile* IP para IPv6 possui uma modalidade com otimização de rota, na qual os pacotes destinados ao nó móvel em uma rede estrangeira poderão ser distribuídos diretamente ao *care-of-address*. Com base nesta particularidade, o *Mobile* IPv6 evita o roteamento triangular.

2.2.2 HIP – *Host Identity Protocol*

Como visto anteriormente, e citado em diversas referências da literatura (BARBATO, 2007), (JIANLI, et al., 2008) e (MOSKOWITZ & NIKANDER, 2006), o padrão *Mobile* IP não resolve totalmente os problemas de mobilidade e segurança na Internet, pois ele confia totalmente no sistema de roteamento. Um usuário mal intencionado pode se passar por outro e efetuar ataques de negação de serviço (DoS – *Denial of Service*), por exemplo, através de mensagens falsas de atualização de endereço.

De acordo com Moskowitz e Nikander em (MOSKOWITZ & NIKANDER, 2006), existem três falhas críticas no *namespace* atual. Em primeiro lugar, o reendereçamento dinâmico não pode ser gerido diretamente; em segundo lugar, o anonimato não pode ser fornecido de maneira consistente e confiável; finalmente, não é fornecida autenticação para sistemas e pacotes. Estas deficiências decorrem do fato de que as plataformas de computação atuais nomeiam de forma ineficiente ao utilizar o *namespace* atual.

Contudo, outras propostas têm sido estudadas. O HIP (RFC-4423) (MOSKOWITZ & NIKANDER, 2006) é um protocolo alternativo ao *Mobile* IP e é baseado na criação de um novo *namespace*, que fornece um nome estático ao *host* para a identificação exclusiva do mesmo, deixando o endereço IP somente para a localização do *host* na topologia da rede.

Ainda de acordo com Moskowitz e Nikander em (MOSKOWITZ & NIKANDER, 2006), a principal ideia do HIP é criar um novo espaço de nomes entre as camadas de rede e de transporte na arquitetura de protocolos da Internet atual. Essa nova camada, a camada de identificação de *host*, utiliza um identificador de *host* (HI – *Host Identifier*) para identificar os nós em rede e criar uma ligação dinâmica com o respectivo localizador (endereço IP), ou seja, a camada de identificação de *host* corresponde a um ponto de indirecionamento entre o identificador HI e o localizador de *host* (endereço IP).

A comunicação entre *hosts* utilizando o HIP não é atrelada à dupla semântica do endereço IP, permitindo que um *host* seja identificado unicamente nas camadas de aplicação e transporte através do novo *namespace* e localizado através do endereço IP. Resumidamente, o

HIP não utiliza o endereço IP como identificador de nós, já que separa as camadas de rede e de transporte, permitindo o deslocamento de um nó móvel sem perder as conexões ativas.

O HI é estático e único globalmente e apesar desta identidade ser associada com a pilha de protocolos IP, existe a possibilidade da utilização em outras pilhas de protocolos. Esta característica torna o HIP uma solução bastante interessante para arquiteturas pós ou não-IP. Além disso, cada HI é associado unicamente a um *host* e utiliza conceitos criptográficos para a sua definição. O objetivo de se utilizar a criptografia para criar identificadores é a possibilidade de realizar uma autenticação em conexões estabelecidas em meios não-confiáveis. Além disso, a criptografia baseada em chaves públicas permite que cada nome seja considerado estatisticamente único em um ambiente global.

A **Figura 5** ilustra parte da pilha de protocolos do IP (lado esquerdo) em contraste com a nova pilha de protocolos do HIP (lado direito). Nesta última, o identificador do *host* e seu localizador são separados um do outro, sendo que o endereço IP continuará atuando como localizador, enquanto o HI é responsável por identificar o *host* final.

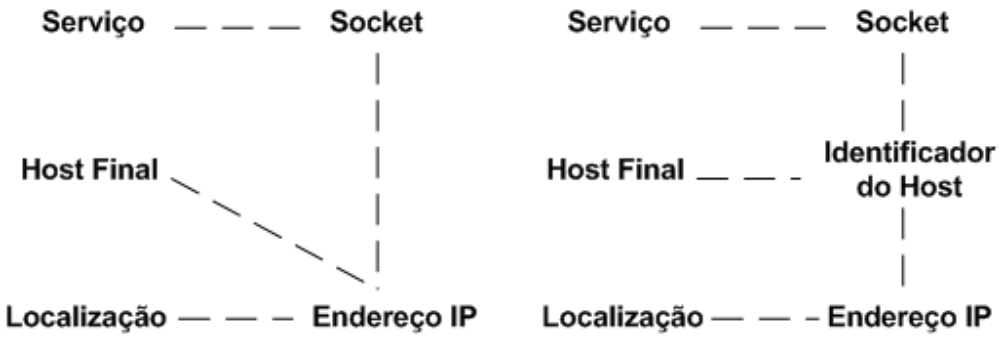

Figura 5. *Internet atual* (*lado esquerdo*)*, protocolo HIP* (*lado direito*) *(MOSKOWITZ & NIKANDER, 2006).*

2.2.3 LISP – *Locator Id Separation Protocol*

O LISP (LEWIS, et al., 2010) é uma proposta da *Cisco Systems* com o objetivo semelhante aos dos protocolos *Mobile* IP e HIP, ou seja, o suporte a mobilidade e *multihoming* para o protocolo IP. Porém, o protocolo LISP é baseado no mapeamento entre endereços de borda e de núcleo e no tunelamento IP sobre UDP (*User Datagram Protocol*) para a entrega de pacotes. Segundo Lewis et al. em (LEWIS, et al., 2010), o LISP é um protocolo utilizado

para implementar a separação do endereço de IP em EIDs (*Endpoint Identifiers*) e RLOCs (*Routing Locators*). Este mecanismo não exige nenhuma alteração nos *hosts* finais, nem mudanças nas infra-estruturas de base de dados existentes.

A implementação do LISP é desenvolvida nos roteadores de borda de uma rede IP, cujo endereço IP é utilizado como localizador de roteamento (RLOC) para os *hosts* do seu domínio. Estes roteadores são responsáveis pelo mapeamento entre os identificadores EIDs e os localizadores dos *hosts* envolvidos na comunicação (IANNONE, et al., 2009) e (BALDO, 2007).

Dado que o domínio de destino já foi determinado pelo ITR (*Ingress Tunnel Router*), este roteador realiza uma pesquisa para mapear um EID em um RLOC a fim de determinar o caminho de roteamento até o ETR (*Egress Tunnel Router*). Os pacotes enviados para o destinatário são encapsulados (um datagrama inserido dentro de outro) no ITR com um novo cabeçalho, onde o campo de destino é o RLOC do destinatário. Esse RLOC é responsável pelo roteamento até o domínio de destino. No domínio do destinatário, o ETR vai desencapsular o pacote e roteá-lo de acordo com o EID do *host* de destino. Este processo cria um túnel entre os roteadores de borda. A **Figura 6** ilustra o funcionamento do LISP.

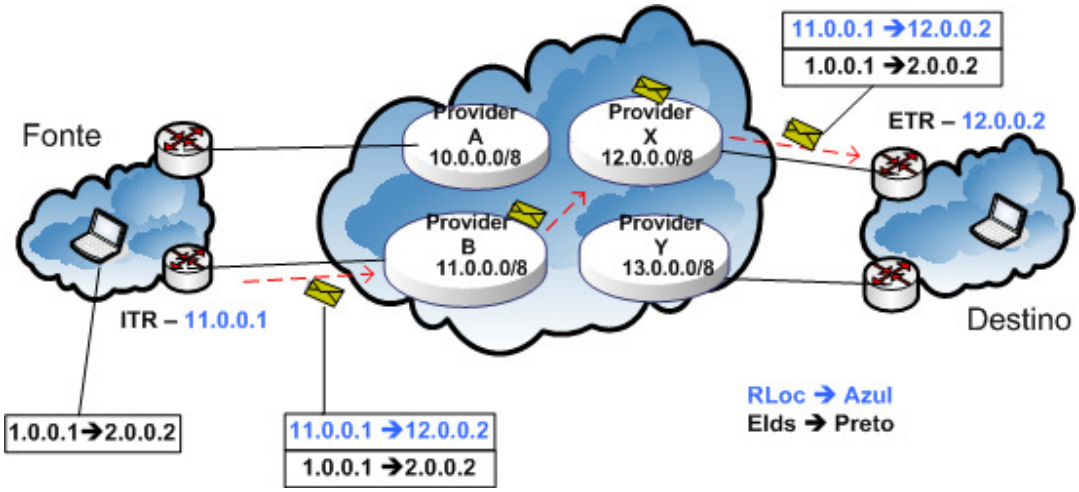

Figura 6. *Funcionamento do LISP (MEYER, 2009).*

Considere o cenário da Figura 6, onde o *NóFonte* (EID = 1.0.0.1) quer comunicar-se com o *NóDestino* (EID = 2.0.0.2). Dado que o ITR (RLOC = 11.0.0.1) escolhido conhece o ETR de destino (RLOC = 12.0.0.2), ele encapsula os dados contendo o EID do *NóFonte* e os envia para o ETR do *NóDestino*. O ETR, por sua vez, recebe os dados desencapsulando-os e encaminha estes dados para o *NóDestino* através do seu EID 2.0.0.2. Em outras palavras, o *NóFonte* conhece o identificador EID do *NóDestino* e o ITR conhece o localizador RLOC do ETR.

Apesar do *overhead* acrescentado pelo encapsulamento e a inflexibilidade para a utilização do LISP em arquiteturas pós ou não IP, existem diversos benefícios alcançados através da separação do espaço de endereçamento atual em EIDs e RLOCs. Dentre eles, tem-se: a redução do tamanho da tabela de roteamento na zona livre *default* (DFZ[2] - *default-free zone*); *multihoming* para *sites* que estão ligados a diferentes prestadores de serviço, nos quais se podem controlar as suas próprias políticas de fluxo e facilidade no reendereçamento quando os clientes mudam de prestador de serviços (LEWIS, et al., 2010).

2.2.4 MILSA – *Mobility and Multihoming Supporting Identifier Locator Split Architecture*

A arquitetura MILSA (JIANLI, et al., 2008) é uma proposta de solução para os problemas de nomeação, endereçamento e roteamento da Internet atual. Existem três princípios adotados na solução MILSA: (i) separação das relações de confiança, denominadas de domínio, das relações de conectividade, denominadas de zona; (ii) separação entre as funções de sinalização e de plano de dados, a fim de melhorar o desempenho e dar apoio à mobilidade; (iii) separação do identificador e localizador fornecendo transparência para as camadas de aplicação e de transporte.

Ainda segundo Jianli (JIANLI, et al., 2008), um domínio representa um grupo de *hosts* de uma mesma hierarquia e é responsável pela atribuição do identificador para as entidades pertencentes a ele. Os domínios de uma mesma hierarquia formam relações lógicas de confiança. A zona por sua vez, é uma unidade agregada topologicamente de uma rede física

[2] DFZ são os roteadores que fazem parte do núcleo da Internet e não possuem a entrada *default* nas suas tabelas, eles são, portanto, um dos mais beneficiados pelo desacoplamento de endereços.

e é responsável por atribuir e agregar topologicamente os endereços dos *hosts* conectados a ela.

A ligação lógica entre o domínio e a zona é feita pelo RZBS (*Realm Zone Bridging Server*). Este servidor pode ser concebido na forma hierárquica, acomodando a estrutura de um dado domínio. Em outras palavras, uma autoridade do domínio se encarrega de identificar os *hosts* pertencentes logicamente a ele, enquanto uma autoridade da zona mantém as informações de um ou mais endereços ou localizadores destes *hosts* e o RZBS se encarrega de interligar o domínio e a zona, mapeando dinamicamente os identificadores aos localizadores dos *hosts*. **A Figura 7** ilustra a arquitetura MILSA.

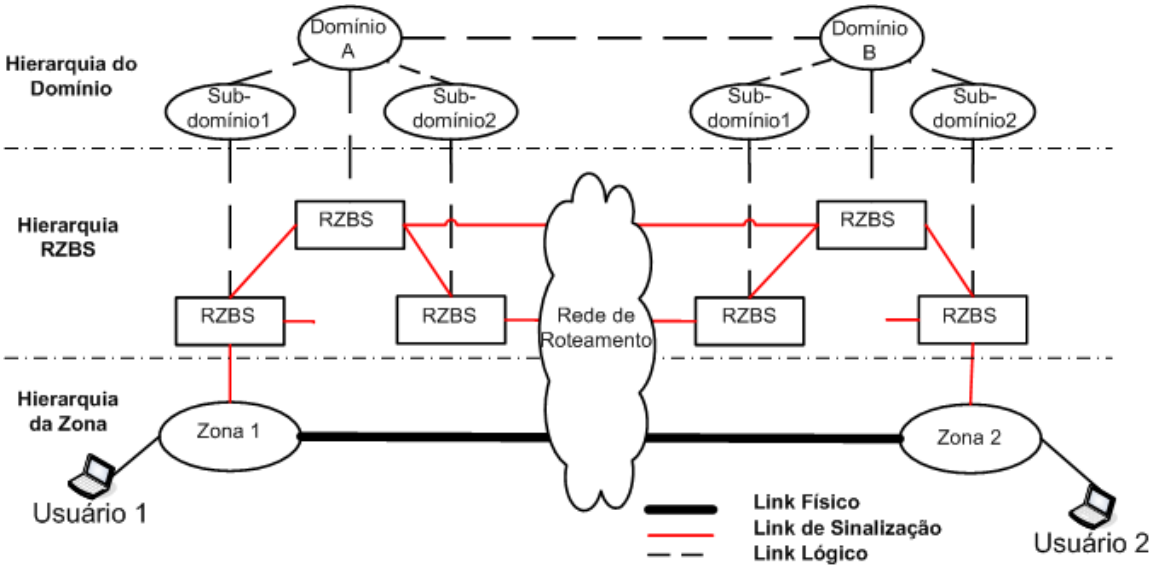

Figura 7. *Arquitetura conceitual MILSA* (*JIANLI, et al., 2008*).

Na concepção do MILSA, os identificadores dos usuários dos dois terminais na Figura 7 poderiam ser "Usuario-1.Subdominio-1.Dominio-A" e "Usuario-2.Subdominio-2.Dominio-B" respectivamente, dos quais, a parte mais a esquerda do identificador seria concebido na forma plana e o restante do nome concebido na forma hierárquica para representar a posição lógica na hierarquia do domínio. A **Figura 8** exemplifica a formação de um nome para a arquitetura MILSA.

Figura 8. *Exemplo de formação do nome MILSA.*

A parte plana do nome tem que ser única no subdomínio para evitar conflito de nomes e pode ser criada com base em criptografia de chave pública ou algoritmos *hash.* Se ambos os usuários estiverem no mesmo subdomínio, não há necessidade de utilizar os nomes completos, é necessário, portanto, apenas a parte mais à esquerda do nome.

Na arquitetura MILSA, um identificador hierárquico do tipo URI (*Uniform Resource Identifier*) (HUI – *Hierarchical URI-like Identifier*) é utilizado pelas camadas de aplicação e de transporte, sendo mapeado dinamicamente para um localizador que corresponde à posição atual do nó na topologia de rede. Este mapeamento é feito pela subcamada HMS (HUI *Mapping Sublayer*) através da interação com a infra-estrutura RZBS. O HMS, portanto, oferece suporte à mobilidade, já que existe a separação do identificador e do localizador e suporte ao *multihoming*, uma vez que um identificador pode estar ligado a vários localizadores.

Conforme Jianli (JIANLI, et al., 2008), a nova subcamada de mapeamento HMS é introduzida na camada de rede e está localizada entre os cabeçalhos de *host* e o roteamento. O cabeçalho do *host* inclui os cabeçalhos AH (*Authentication Header*) e ESP (*Encapsulating Security Payload*), bem como o cabeçalho de fragmentação/remontagem e o cabeçalho de opção de destino como ilustrado na **Figura 9**.

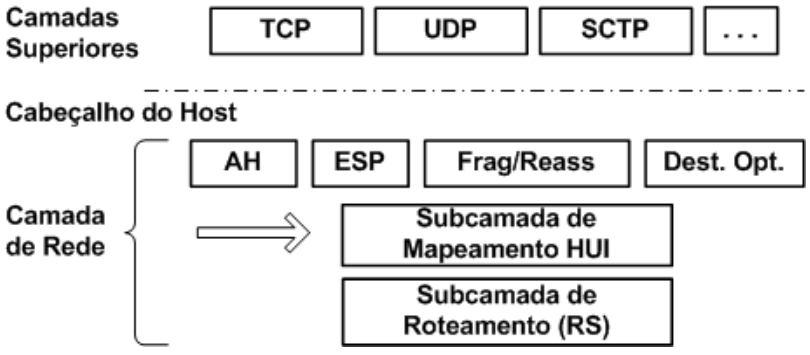

Figura 9. *Acréscimo da sub-camada de mapeamento MILSA* (*JIANLI, et al., 2008*).

O roteamento MILSA é baseado no mecanismo IP atual, embora possa adaptar a qualquer novo sistema de roteamento futuro, por exemplo, o roteamento baseado em nomes planos (ROFL - *Routing on Flat Label*) (JIANLI, et al., 2008) *apud* (CAESAR, et al., 2006). O ROFL (CAESAR, et al., 2006) utiliza rótulos para identificação plana (não-hierárquica) dos nós e desassocia esta identificação da localização dos nós da rede. Este roteamento é baseado em consultas DHTs utilizadas geralmente por protocolos de redes *peer-to-peer*, devido à sua natureza distribuída (CAMPISTA, et al., 2010).

2.2.5 Arquitetura de desacoplamento do projeto Akari

O Akari (HARAI, et al., 2007) é um projeto Japonês com a participação do governo, universidades e o setor privado que visa projetar e implementar uma rede de nova geração até o ano de 2015. O lema do projeto é "uma pequena luz na escuridão que aponta para o futuro" e a sua filosofia é buscar a solução ideal de arquitetura para uma rede de nova geração.

O projeto Akari possui três princípios elementares que embasam a criação da arquitetura de rede de nova geração: o princípio KISS (*Keep It Simple, Stupid*) afirma que a camada de rede deve ser mantida o mais simples possível; o princípio da conexão ao mundo real, que dá suporte à interação do mundo virtual com o mundo real e confirma a necessidade do desacoplamento da identificação e localização; e o princípio da evolução sustentável, que significa que a rede deve se tornar um ambiente livre para evolução e desenvolvimento, sendo capaz de suprir a demanda da sociedade por 50 ou 100 anos (HARAI, et al., 2007).

A proposta de arquitetura do projeto Akari utiliza conjuntos distintos de entidades para identificar e localizar *hosts* em rede. Entretanto, esta proposta é diferente das citadas anteriormente pelo fato de ser independente de tecnologia de interconexão. Em outras palavras, a solução proposta pelo projeto Akari pode ser aplicada em arquiteturas pós IP ou não-IP.

Na arquitetura do projeto Akari os identificadores podem ser concebidos de forma hierárquica ou plana. Os identificadores estabelecidos na forma hierárquica suportam maior

abrangência e escalabilidade, além de prover dicas para a resolução de localizadores. Entretanto, eles podem exigir uma entidade central para atribuir os componentes hierárquicos. Por outro lado, os identificadores planos permitem que os nós criem os seus identificadores de forma autônoma. Os autores do projeto consideram que para ambos os tipos de identificadores, a manutenção de um banco de dados de mapeamento da identificação e localização global é muito importante (HARAI, et al., 2007).

Ainda segundo Harai (HARAI, et al., 2007), um *host* pode ser identificado por duas formas: pelo nome e/ou pelo identificador (ID). Um nome pode ser local ou global. Os nomes locais são únicos na rede local e são usados para identificação do *host* e gerenciamento de rede. Estes nomes são gerados pela combinação de palavras características do *host*, como a sua função em um contexto, proprietário, número de série ou dia e hora da instalação do *host* na rede.

Os nomes globais são formados pela combinação do nome local e do nome IMS (*Identity Management Server*). O IMS é usado para suporte à mobilidade global e segurança no acesso às redes. O nome local gerado é registrado no IMS, onde também é verificada a sua unicidade. Os IDs por sua vez são formados pelo nome global e parâmetros adicionados, usando uma função *hash* criptográfica. Os IDs são inseridos nos cabeçalhos dos pacotes para a função de identificação dos *hosts* finais em uma sessão de comunicação (HARAI, et al., 2007).

A **Figura 10** ilustra a formação do identificador de *hosts* e o mapeamento deste identificador para um localizador.

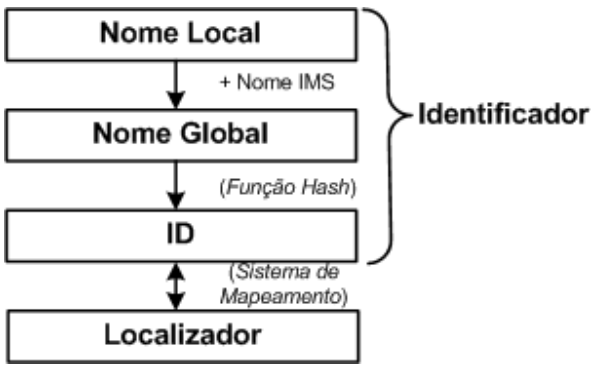

Figura 10. *Formação do identificador e do mapeamento Akari* (*HARAI, et al., 2007*).

Para suportar a arquitetura de desacoplamento e fazer o mapeamento entre identificadores e localizadores dos nós, foi inserida na pilha de protocolos uma nova camada entre a camada de transporte e a camada de rede, denominada de camada de identidade (HARAI, et al., 2007). A inserção da nova camada de identidade e o seu relacionamento com as demais camadas é ilustrada na **Figura 11**.

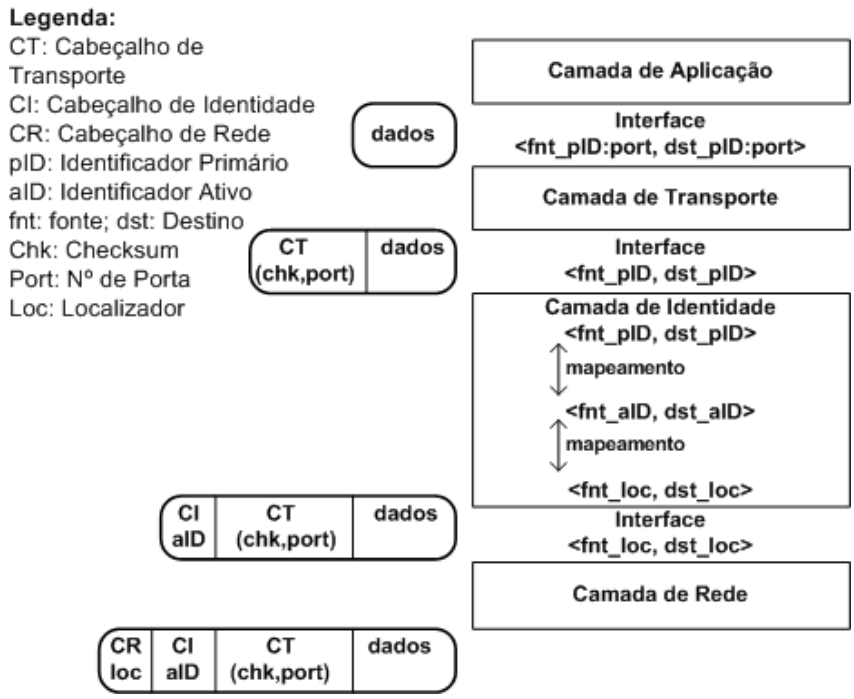

Figura 11. *Inserção da nova camada de identidade Akari (HARAI, et al., 2007).*

Considere a pilha de protocolos da Figura 11. A camada de aplicação envia os dados para a camada de transporte através de uma interface identificada pelos identificadores primários de fonte e de destino, além do número de porta da aplicação. A camada de transporte, por sua vez, insere o cabeçalho de transporte no pacote e o envia para a camada de identidade através de outra interface identificada também pelo identificador primário. Na camada de identidade, este identificador primário é mapeado para um identificador ativo, o qual é inserido no cabeçalho desta camada. Um segundo mapeamento entre o identificador ativo e o localizador dos *hosts* é também desenvolvido na camada de identidade. Então, esta camada insere o identificador ativo no pacote e envia este pacote para a camada de rede,

através de uma interface identificada pelos localizadores de fonte e de destino. Em seguida, os localizadores de fonte e de destino são inseridos no cabeçalho da camada de rede e o pacote é então enviado para o seu destino.

Conforme Harai (HARAI, et al., 2007), os identificadores ativos inseridos pela camada de identidade e os localizadores inseridos pela camada de rede são visíveis por toda a rede. E ainda, em uma determinada sessão de serviço, o identificador ativo permanece estático enquanto o localizador pode ser modificado dinamicamente para o suporte à mobilidade, *multihoming* e engenharia de tráfego.

2.3 Comparações entre *Mobile* IP, HIP, LISP, MILSA e Akari

A escolha da arquitetura de nomeação é um importante ponto de partida no *projeto* de uma arquitetura de rede, já que vários aspectos, como o de segurança e de roteamento, são dependentes da forma como os nomes são concebidos. Para exemplificar, a criptografia de chave pública e o sistema de roteamento ROFL são abordagens de segurança e roteamento citadas para nomes planos, enquanto para nomes hierárquicos são citados o IPSec (*Internet Protocol Security*) e o roteamento DNS atual, para aspectos de segurança e roteamento, respectivamente.

O Mobile IP e o LISP utilizam a arquitetura de nomes hierárquicos atual, sendo que ambos dividem o endereço IP em dois espaços de nomes hierárquicos para suportar o desacoplamento da identificação e localização dos *hosts*. O HIP utiliza um espaço de nomes planos para identificar unicamente o *host* e o endereço IP para a localização do *host* na topologia da rede. Por outro lado, na arquitetura MILSA e no projeto Akari os identificadores de *hosts* utilizam parte do nome plano para critérios de segurança e parte hierárquica, visando melhorar a escalabilidade das propostas.

De acordo com Harai (HARAI, et al., 2007), a maioria destas abordagens é baseada em identificadores inflexíveis (utilizando endereço IP), como o *Mobile* IP e o LISP, ou baseada em identificadores gerados por criptografia de chave pública, como o HIP. A vantagem de se utilizar identificadores baseados em IP é que os aplicativos criados para a Internet atual podem continuar sendo utilizados sem a necessidade de alterações. Entretanto,

estas abordagens são inflexíveis e não podem, portanto, ser utilizadas em arquiteturas pós ou não-IP. Já os identificadores baseados em criptografia de chave pública, apesar de serem seguros, são longos e ilegíveis para humanos.

Por outro lado, os identificadores da arquitetura MILSA são baseados em IP, mas podem ser adaptados para utilizar outro tipo de protocolo. Já os identificadores utilizados pelo projeto Akari são totalmente flexíveis, independentes da tecnologia de interconexão, portanto. Além disso, estes identificadores são criados com base na função *hash* do nome do *host*, sendo que estes nomes são criados de forma legível e são compostos em nível local e global.

Como suporte de segurança o *Mobile* IP utiliza o IPSec, e para o LISP a segurança é baseada no processo de mapeamento de EIDs para RLOCs. Os modelos HIP, MILSA e Akari utilizam o conceito de identidade criptográfica para cifrar as informações como forma de implementação de segurança para a transmissão de pacotes.

O *Mobile* IP não oferece suporte transparente à mobilidade, ou seja, a atualização da localização do nó móvel é mapeada pelo agente domiciliar ao invés de ser transmitida diretamente entre os *hosts*. Este fato implica em longos períodos de espera no processo do registro de atualização e pode ocasionar perda de pacotes, além do problema de roteamento triangular citado anteriormente. A otimização de roteamento para o *Mobile* IPv6 tenta resolver este último problema, mas requer mudanças consideráveis para ambos os *hosts* finais (JIANLI, et al.,2008).

A abordagem LISP possui alguns pontos negativos, tais como: o aumento do *overhead* e de atrasos causados pelo mapeamento dos EIDs e RLOCs, o que pode causar atrasos na transmissão e perda de pacotes. Enquanto para o HIP, o aumento de *overhead* acontece na camada de identificação de *host* e a perda de pacotes pode acontecer quando os dois terminais da comunicação se movem ao mesmo tempo.

Para o *Mobile* IP, MILSA e Akari o aumento de *overhead* acontece devido ao encapsulamento dos dados ao tunelar os pacotes do agente local para o agente estrangeiro, na camada HMS e na camada de identidade, respectivamente. Outro problema conhecido do *Mobile* IP é o uso de pacotes de atualização de endereços durante ataques de negação de

serviço, ou seja, um usuário mal-intencionado pode enviar pacotes falsos de atualização de endereço correspondente a um *host* que não seja o seu (BARBATO, 2007).

A **Tabela 1** resume as comparações entre as características dos protocolos de desacoplamento de identificadores e localizadores estudados.

Tabela 1. *Algumas comparações entre as arquiteturas de desacoplamento de identificadores e localizadores de hosts.*

	***Mobile* IP**	**HIP**	**LISP**	**MILSA**	**Akari**
Arquitetura de Nomes	Hierárquica (IP); utiliza nomes legíveis.	Plana; utiliza nomes opacos.	Hierárquica (IP); utiliza nomes legíveis.	Parte plana e parte hierárquica com nomes legíveis.	Plana, parte hierárquica com nomes legíveis locais e globais.
Flexibilidade do Roteamento	Apenas IP – Inflexível.	IP, pós-IP ou não-IP – Flexível.	Apenas IP – Inflexível.	Roteamento baseado em IP, futuramente pode usar ROFL – Pode adaptar-se para se tornar flexível.	Totalmente flexível, roteamento independente de tecnologia de interconexão.
Segurança na Identificação	IPSec; Não contém estruturas de chave pública ou outras entidades de certificação.	Baseado em criptografia de chave pública, problemas com ataques de negação de serviços.	Estritamente no mapeamento EID-RLOC. Sem estruturas de chave pública ou outras ent. de certificação.	Baseado em criptografia de chave pública.	Baseado em criptografia de chave pública e função *hash.*
Questões de desempenho	Aumento de *overhead* (encapsulamento); roteamento triangular, espera nos registros de atualização.	Aumento de *overhead* – na camada de identificação de *host.*	Aumento de *overhead* devido ao mapeamento; Latência nas pesquisas EID-RLOC.	Aumento de *overhead* – na camada HMS.	Aumento de *overhead* – na camada de identidade.
Perda de Pacotes	Devido aos longos períodos de espera no processamento do registro da atualização.	Quando dois terminais se movem ao mesmo tempo.	Pode acontecer devido aos atrasos de mapeamento.	_____	_____

Resumidamente, um evento de mobilidade ou de *renumbering* leva a uma atualização do localizador do *host* e a um mapeamento do identificador neste localizador atualizado, desenvolvido conforme uma arquitetura.

O *Mobile* IP atualiza o *care-of-address* do nó móvel junto ao agente domiciliar. Já o HIP mapeia o identificador HI no localizador atualizado na camada de identificação de *host*, enquanto o LISP atualiza o localizador da nova rede de borda através do mapeamento EID-RLOC. O MILSA, por sua vez, desenvolve este mapeamento dinâmico através dos servidores RZBS. E finalmente, para o projeto Akari, o mapeamento do identificador ativo e o localizador atualizado acontece na camada de identidade.

O desacoplamento entre a identificação e a localização de *hosts* é uma das soluções importantes para lidar com as deficiências de suporte à mobilidade, *multihoming*, dentre outros problemas decorrentes da dupla funcionalidade do endereço IP. Embora hoje existam várias propostas para a separação da identificação e localização de dispositivos em rede, conforme Jianli (JIANLI, et al., 2008), a maioria delas não fornece uma solução completa para o relacionamento de nomes e endereços no contexto conjunto de mobilidade, *multihoming* e segurança.

2.4 Desacoplamento da identificação e localização da informação – Redes Centradas na Informação

Nas seções seguintes será dada uma visão geral sobre três abordagens de redes centradas na informação, bem como algumas comparações das características mais relevantes destas abordagens e a relação com o desacoplamento de identificação e localização da informação. É importante salientar que apesar das particularidades, os três projetos a serem citados adotam o modelo de comunicação publica/assina (*publish/subscribe*), na qual o interessado em uma determinada informação deve solicitá-la a rede. Este modelo visa trazer ao receptor o controle do fluxo de comunicação, saindo, portanto, do modelo "receptor aceita tudo" atual. Os termos "centrado no conteúdo" e "orientado à informação" são aplicados essencialmente da mesma forma para indicar redes nas quais os objetos de informação em si são o foco principal, em vez dos nós da rede. A seguir, portanto, os termos informação e conteúdo serão utilizados de forma equivalente.

2.5 4WARD NetInf – Network of Information

O 4WARD (*Architecture and Design for the Future Internet*) é um projeto de Internet do futuro patrocinado pelo FP7 (*European Union Framework Programme* 7) e visa criar uma nova proposta de arquitetura de rede baseada na abordagem *clean slate*. O projeto 4WARD é dividido em seis grupos de trabalho incluindo o NetInf (NIEBERT, et al., 2008).

A ideia principal do NetInf é obter um sistema para a localização e recuperação da informação baseado em um identificador único desta informação. Para tal, é necessário um mecanismo de resolução de nomes que faça o mapeamento de um identificador para um localizador desta informação. A proposta considera que este mecanismo deve ser persistente e independente de cópia ou localização. Em outras palavras, o identificador de um objeto de conteúdo não pode mudar ao se relacionar com uma cópia ou com a mudança do localizador deste objeto.

Além disso, o sistema de nomeação para os objetos de informação do NetInf é concebido de forma plana, ou seja, não hierárquica. Desta forma, este mecanismo plano é incompatível com a concepção hierárquica do DNS atual. Apesar do mecanismo plano não ser legível a humanos, este mecanismo apóia questões de segurança e privacidade e não exige entidades centralizadas de gerenciamento de nomes.

Para Dannewitz (DANNEWITZ, 2009), o NetInf estende o conceito de desacoplamento da identificação e localização com outro nível de indireção, visando desacoplar os objetos de informação de seus locais de armazenamento, ou seja, criar acesso à informação independente do local de armazenamento e se beneficiar de cópias distribuídas ao longo da rede.

O modelo de informação do NetInf é baseado em dois objetos: o objeto de informação (IO – *Information Object*) e o objeto de dados (DO – *Data Object*) ou objeto binário (BO – *Binary Object*). Os IOs são objetos que representam a informação propriamente dita. Eles podem fazer referência a uma imagem, um som, um conteúdo de vídeo, páginas *Web* e assim por diante. Já os DOs são um tipo especial de IO que representam um objeto no seu nível de *bit*. Um DO representa, por exemplo, a seqüência de *bits* de um arquivo, uma

seqüência binária resultante da codificação de mídia, tal como um arquivo com codificação MP3, ou um texto.

A partir deste modelo de informação do NetInf, a pesquisa e recuperação da informação pode se desenvolver de forma mais eficiente. Uma pesquisa baseada em um IO pode, por exemplo, referir-se de forma geral a uma imagem ou uma música sem especificar a codificação desta imagem ou música, dado que muitas vezes estas informações não são relevantes ao usuário. Os IOs, portanto, possibilitam que os usuários localizem o conteúdo interessado independentemente de sua representação específica ou outras características que, a princípio, não são interessantes a esse usuário.

Segundo (AHLGREN, et al., 2010), a pesquisa de objetos de informação do NetInf pode ser feita através de um mecanismo de busca integrado na arquitetura NetInf, ou alternativamente, através de um mecanismo de busca externa. Ambos, porém, devem incluir novos tipos de funcionalidades baseadas em atributos do mundo real de entidades físicas, como posição GPS (*Global Positioning System*), etiquetas RFID (*Radio Frequency Identification*) e reconhecimento de imagens do mundo real.

O paradigma publica/assina (*publish/subscribe*) do NetInf é implementado a partir da publicação de um objeto de informação. Esta publicação é feita ao registrar o nome do objeto de informação em um sistema de resolução de nomes, e a assinatura por sua vez, é feita quando algum interessado solicita ao sistema de resolução de nomes algum objeto publicado. A localização e assinatura de um objeto de informação no NetInf só é possível, portanto, depois que este for publicado.

Para que se tenha segurança e confiança no conteúdo a ser disseminado na rede, o NetInf propõe que a informação seja auto-certificável, ou seja, a confiança deve ser nativa do conteúdo, ao contrário das redes de hoje em que a confiança é baseada no *host* que enviou este conteúdo. A auto-certificação permite verificar a integridade dos dados independentemente do *host* que enviou o conteúdo ou do canal de transmissão. Com base no identificador, o usuário pode verificar se recebeu os dados corretos e se os dados não foram alterados, ou seja, se os dados estão íntegros (OHLMAN, et al., 2009).

A autenticação do proprietário, por sua vez, é outra característica importante do esquema de nomeação NetInf (OHLMAN, et al., 2009). Ela permite tirar conclusões não só sobre a integridade dos dados, mas também sobre a qualidade do conteúdo e a confiabilidade do seu publicador. A integridade do conteúdo é garantida através de uma ligação criptográfica entre o objeto de informação e o nome deste objeto e a autenticidade é garantida usando um *hash* de chave pública como parte do nome auto-certificável. A verificação da autenticidade, portanto, necessita de uma relação com uma entidade externa que forneça a chave privada correspondente.

Conforme Alhgren (AHLGREN, et al., 2010) e Dannewitz (DANNEWITZ, 2009), os nomes da arquitetura NetInf são constituídos de três campos: um campo *Tipo* responsável por definir o formato do nome, que especifica, por exemplo, o algoritmo *hash* utilizado no processo de auto-certificação; um campo *Autenticador*, responsável por fazer a ligação entre o identificador do IO à chave pública; e finalmente, um campo *Rótulo*, que contém um número arbitrário de etiquetas identificadoras do objeto de informação. A combinação do *Autenticador* com o *Rótulo* necessita ser único globalmente.

A arquitetura NetInf faz o uso de DHT para o roteamento dos conteúdos. Segundo (LIU, et al., 2009), as DHTs são uma classe de sistemas distribuídos descentralizados que prestam um serviço de pesquisa semelhante a uma tabela *hash*. Alguns algoritmos DHTs, bem como a sua estrutura e outros benefícios da sua utilização serão apresentadas no Capítulo 3.

2.5.1 Esquema de nomeação NetInf

Na arquitetura NetInf, o esquema de nomeação é projetado para ser flexível e genérico, podendo nomear qualquer tipo de entidade. Os IOs (*Information Objects*) são uma espécie de entidade virtual que pode representar objetos genéricos, estáticos ou dinâmicos, nós de rede, pessoas, lugares ou outros objetos do mundo real, além de ser flexível para integrar outras propriedades, como as de segurança. Ao mesmo tempo, o esquema de nomeação fornece identificadores persistentes, ou seja, independentes da localização do conteúdo, do proprietário deste conteúdo, ou da estrutura organizacional do proprietário (DANNEWITZ, et al., 2010).

Para evitar a inconsistência e redundâncias desnecessárias no sistema de nomes, cópias idênticas de entidades devem ter nomes idênticos, contendo apenas informações relevantes ao conteúdo e não o local de armazenamento. Não deve haver qualquer diferença entre o nome do objeto original e o nome de cópias do mesmo objeto. Para isto, o local de armazenamento não pode influenciar semanticamente na nomeação dos conteúdos. Desta forma, a arquitetura de nomeação precisa suportar naturalmente o desacoplamento da identificação e localização dos conteúdos nomeados.

Na arquitetura NetInf existe o desacoplamento entre os publicadores e os assinantes da informação. A interface de programação de aplicações API (*Application Programming Interface*) tem o estilo publica/assina (*publish/subscribe*), onde os autores publicam o conteúdo e os clientes assinam este conteúdo, sem que exista a necessidade deles se conhecerem ou de confiarem entre si. Qualquer *host* ou servidor que possuir uma cópia do conteúdo requisitado pode, portanto, enviá-la para o assinante. Além disso, a API do NetInf fornece pesquisa de objetos de informação, bem como o mapeamento de identificadores em localizadores de informação (AHLGREN, et al., 2010).

O esquema de nomeação proposto pelo NetInf suporta a combinação de persistência, auto-certificação e autenticação do proprietário (DANNEWITZ, 2009 & OHLMAN, et al., 2009). A persistência significa que os nomes devem sempre se manter válidos, mesmo ao alterar o local de armazenamento, as informações do proprietário, o prestador de serviço, os algoritmos utilizados para nomear a informação (por exemplo, o algoritmo de *hash* utilizado), sobretudo, o próprio conteúdo. Para conteúdos dinâmicos como uma página *Web*, mesmo quando o seu conteúdo for modificado, os nomes deverão permanecer válidos para serem usados como links para esta página da *Web*.

2.5.2 Suporte à mobilidade NetInf

O suporte à mobilidade é nativo da arquitetura NetInf. Em (AHLGREN, et al., 2010), é dito que as entidades (equipamentos de usuário final, redes, nós e objetos, como IOs e BOs) podem se mover sem que exista prejuízo à troca de informações. A localização do publicador e do assinante destas entidades é determinada na inicialização da sessão, de forma a utilizar as informações mais recentes da topologia da rede. Quando a topologia muda devido a um evento de mobilidade ou *re-homing*, as entidades afetadas pela mudança atualizam seus IOs em ARs (*Attachment Registers*) que possibilitam a construção de novas localizações válidas para a nova topologia.

Conforme (AHLGREN, et al., 2010), como o NetInf não restringe a utilização de qualquer tecnologia de interconexão, o suporte a mobilidade do equipamento do usuário final (UE – *end-User Equipment*) pode ser desempenhado assumindo arbitrariamente um tipo de endereço, incluindo o IP. A mobilidade em *hosts* pode ser suportada através de algum mecanismo que implemente o desacoplamento entre a identificação e a localização do *host*, como o *Mobile* IP.

A mobilidade em *hosts* é suportada a partir da combinação de três mapeamentos contidos no dicionário NetInf. O primeiro mapeamento permite a ligação entre o identificador do UE, neste caso, desempenhado pelo NAI (*Network Access Identifier*) e o identificador do nó de acesso, <NAI_UE→ ID_NetInf Access Node>. O segundo permite localizar o endereço IP do nó onde o UE se conectou primeiramente <ID_NetInf Access Node → NetInf Access Node Location>. O terceiro é desempenhado pelo nó de acesso NetInf e permite o mapeamento do NAI em um endereço do UE na rede visitada, o *care-of address*, <NAI → CoA>. A **Tabela 2** resume os mapeamentos para suporte à mobilidade NetInf e seus respectivos significados.

Tabela 2. *Dicionário de mapeamentos NetInf.*

Mapeamento	**Significado**
<NAI_UE→ ID_NetInf Access Node>	Mapeamento entre o identificador do UE e o identificador do nó de acesso.
<ID_NetInf Access Node → NetInf Access Node Location>	Mapeamento entre o identificador e o localizador do nó de acesso.
<NAI → CoA>	Mapeamento entre o identificador do UE e o seu localizador na rede visitada.

A **Figura 12** mostra um cenário em que um equipamento móvel representado por UE_2 é conectado inicialmente ao nó de acesso 1 com o mapeamento <NAI_{UE2},ID_NetInfNode1>. Devido a um evento de mobilidade, o UE muda para o nó de acesso 3 e o seu novo mapeamento para a rede visitada passa a ser <NAI_{UE2},ID_NetInfNode3>. O identificador da rede de acesso e o localizador do UE na rede estrangeira (CoA), são atualizados para a rede de acesso 3.

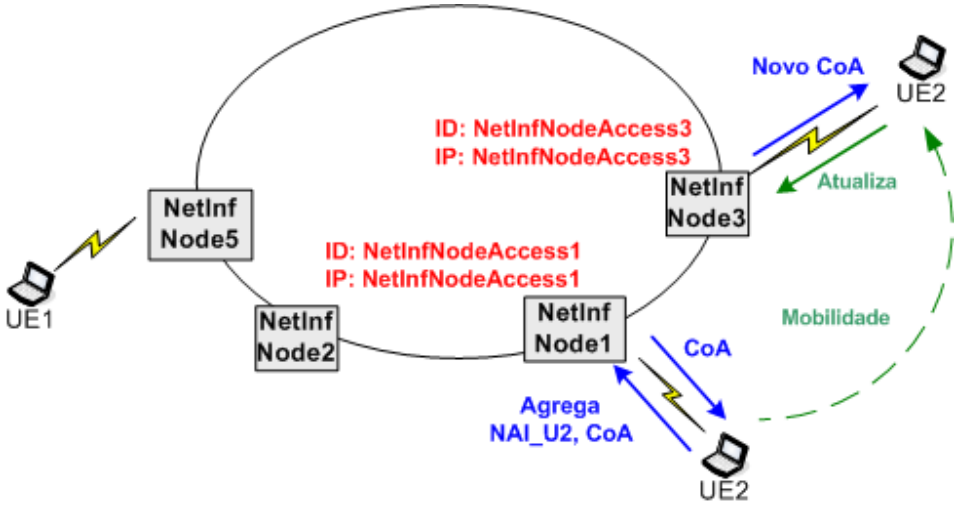

Figura 12. *Mobilidade em NetInf (AHLGREN, et al., 2010).*

É notável que estes mapeamentos sejam semelhantes aos utilizados na arquitetura do *Mobile* IP, sendo que os dois primeiros mapeamentos utilizados na arquitetura NetInf, <NAI_UE→ ID_NetInf Access Node> e <ID_NetInf Access Node → NetInf Access Node Location> são correspondentes ao mapeamento utilizado entre o identificador e o localizador do dispositivo móvel dentro do agente domiciliar (*home agent*), e o terceiro mapeamento, o <NAI → CoA> é feito entre o identificador afixado na rede domiciliar e o endereço (dinâmico) na rede visitante. Note, portanto, que em ambas as arquiteturas, existem um identificador estático, o NAI e o *home address,* para o NetInf e o *Mobile* IP, respectivamente, e o endereço dinâmico denominado de *care-of address,* utilizado nas duas arquiteturas.

2.5.3 Roteamento NetInf

Conforme Ahlgren (AHLGREN, et al., 2008), a recuperação e entrega do conteúdo em NetInf se resume na combinação de dois processos: o de resolução de nomes, que se responsabiliza por prover a localização para um determinado identificador de objeto através de consulta em uma DHT; e o processo de roteamento, que se responsabiliza por encaminhar a requisição para a localização do armazenamento e retornar o conteúdo.

Existem duas abordagens para lidar com os dois processos citados: a primeira alternativa separa a fase de resolução de nomes da fase de roteamento, como é feito tradicionalmente nos esquemas de roteamento baseado em topologia, tal como o OSPF (*Open Shortest Path First*) e o BGP (*Border Gateway Protocol*); a segunda alternativa é um esquema integrado de roteamento baseado em nome, que combina o caminho de resolução e o caminho de recuperação em um processo único.

A primeira abordagem não é animadora, já que resultados de pesquisas em roteamento mostram que redes cujas topologias são dinâmicas não conseguem atingir boa escalabilidade, dado que a complexidade da comunicação cresce proporcionalmente ao aumento elevado do número de nós (DANNEWITZ, 2009). A segunda alternativa investiga o uso de roteamento baseado em nome, como o LLC (*Late Locator Construction*) (AHLGREN, et al., 2008), que integra os caminhos de resolução e de recuperação, podendo resultar em um melhor desempenho. Esta abordagem executa o roteamento através do mapeamento direto entre o identificador do objeto de dados e a rota, o que diminui a latência e resulta, provavelmente, em melhor desempenho (AHLGREN, et al., 2008).

A arquitetura LLC separa o roteamento entre a rede de núcleo estática, que pode ser uma rede legada IPv4 ou IPv6, isto é, empregando qualquer protocolo de roteamento inter-domínio, como o BGP; e as regiões dinâmicas da borda da rede. A introdução do LLC em regiões de borda, portanto, alivia o problema de escalabilidade no roteamento associado ao uso de *sites multihoming* e o uso de prefixos independentes de provedor (OHLMAN, et al., 2009).

Conforme Ohlman (OHLMAN, et al., 2009), na arquitetura LLC, o localizador de um objeto é construído dinamicamente baseado no caminho seguido pelos primeiros pacotes de controle enviados. O assinante consulta o localizador do nó publicador no sistema de resolução de nomes e envia pacotes de controle para o publicador, a fim de levantar informações atualizadas da topologia da rede. A localização do objeto, portanto, é baseada no caminho seguido até a fonte da informação e é construída no último momento antes de enviar os pacotes, daí o termo *Late Locator Construction*. Um localizador baseado em caminho para um determinado objeto, portanto, consiste de uma seqüência de identificadores que descreve o caminho de interconexão ao atravessar uma seqüência de redes de borda em busca do *host* que está associado ao objeto de destino, e finalmente, o objeto de destino.

Em outras palavras, cada entidade de rede como um *host*, um IO, uma rede móvel, ou roteador de borda registra o seu nome nos registros de ligação ARs vizinhos. Os ARs estão distribuídos ao longo da rede e podem ser alcançados através de um sistema de resolução de nomes que faça o mapeamento do identificador do AR para um localizador. Estes registros são consultados de forma seqüencial, a fim de construir o caminho para o localizador do objeto de destino.

Ainda segundo Ohlman (OHLMAN, et al., 2010), dado que o nome do objeto de informação a ser localizado é baseado em um identificador criptográfico, uma DHT pode ser usada para implementar a função do AR. O nome do objeto é usado como uma chave para uma consulta a uma tabela *hash*, e a resposta desta consulta retorna o localizador do AR desejado. **A Figura 13** reproduzida de (AHLGREN, et al., 2010) ilustra a formação do caminho de entrega.

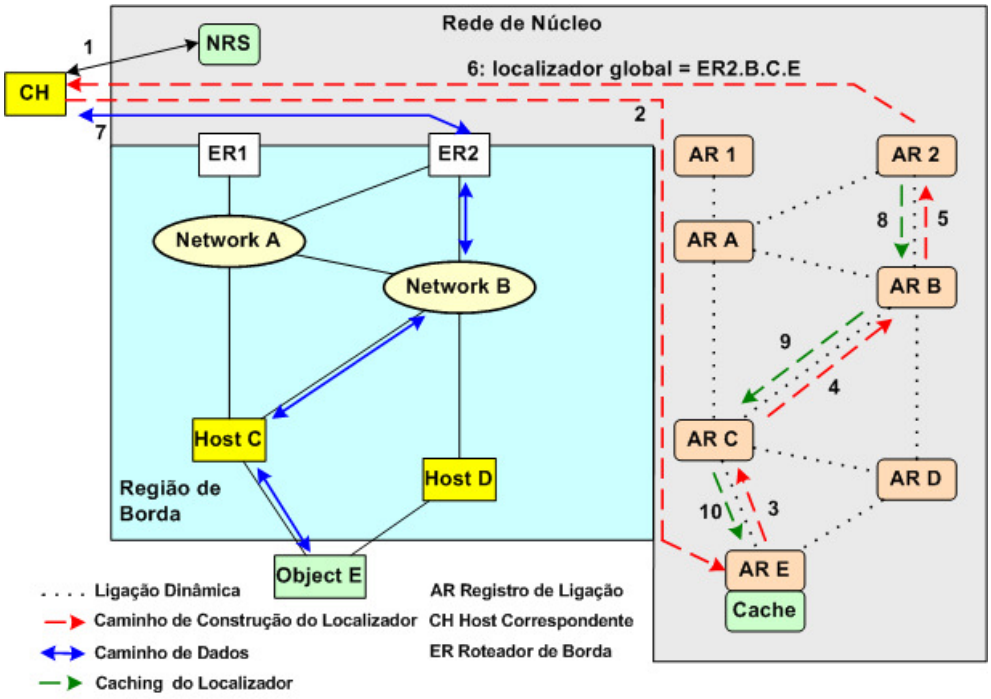

Figura 13. *Formação do caminho de entrega NetInf* (*AHLGREN, et al., 2010*).

Ao iniciar uma sessão, um *Host Assinante* (CH - *Correspondent Host*) solicita ao sistema de resolução de nomes (passo 1 da Figura 13) o localizador de um objeto de destino *AR E*. A consulta retorna o localizador do *AR E*, o qual é responsável por manter informações sobre o nó *cache* onde está armazenado o *Objeto E*. Baseado nas informações de ligação existentes nos ARs, uma seqüência de ARs é então consultada (passo 2 ao 5) para construir o localizador global do objeto de interesse. O localizador do *Objeto E*, poderia ser, por exemplo, *ObjectE@HostC@NetworkB@ER2*. O localizador construído é então devolvido ao *Host Assinante* no passo 6. Este localizador é inserido no cabeçalho dos pacotes enviados pelo *Host Assinante* ao *Host Destino* (*Publicador*) (passo 7) (OHLMAN, et al., 2010).

2.6 CCN – Content-Centric Network

CCN é um projeto do PARC (*Palo Alto Research Center Incorporated*) que visa transformar a forma como as redes de comunicação de dados atuais operam, através de uma mudança na arquitetura de rede que faça a recuperação do conteúdo pelo seu nome, e não pela

sua localização. Neste contexto, a principal motivação das CCNs é melhorar a eficiência da utilização da Internet atual, direcionando o foco da rede para a distribuição de conteúdo.

Desta forma, as CCNs propõem um novo mecanismo de distribuição de conteúdo com os mesmos princípios de engenharia do IP, porém, utiliza a nomeação de conteúdo, em vez de identificadores de *hosts* como foco principal. Segundo Van Jacobson em (PALO ALTO RESEARCH CENTER INCORPORATED, 2010), a visão das CCNs é reutilizar os elementos bem sucedidos do TCP/IP, e construir uma nova rede, substituindo o modelo IP centrado em *hosts* por um modelo orientado a conteúdo, para ser utilizado como o protocolo central de interconexão de redes.

Conforme Jacobson (JACOBSON, et al., 2009), nas CCNs existem dois tipos de pacotes: o de interesse e o de dados. Um consumidor envia por *broadcast* a requisição do conteúdo interessado através de um pacote de interesse. Esta informação fica armazenada em uma tabela de interesses pendentes - PIT (*Pending Interest Table*). Uma base de informação de encaminhamento - FIB (*Forwarding Information Base*) é utilizada para encaminhar pacotes de interesse da tabela PIT, em busca de uma potencial fonte (nó publicador ou *cache*) para a informação desejada.

Qualquer nó que tenha o conteúdo de mesmo nome do pacote de interesse enviado pelo consumidor pode responder com um pacote de dados. Para que uma potencial fonte consiga identificar e localizar o assinante do conteúdo, os pacotes de interesse deixam, figurativamente falando, “migalhas de pão” ao longo do caminho, para que os pacotes de informação retornem pelo sentido reverso do caminho. Desta forma, é possível maximizar o compartilhamento de informações na rede, já que cada pacote enviado pode ser recuperado por vários consumidores interessados. Esta característica reduz os custos de operação, pois uma única cópia pode ser compartilhada por vários assinantes (JACOBSON, et al., 2009).

O controle de fluxo nas CCN é baseado numa regra simples na qual um pacote de interesse pode recuperar no máximo um pacote de dados. Em outras palavras, pacotes de dados e de interesse utilizam uma razão de um para um, o que mantém o controle de fluxo balanceado. Um controle de fluxo semelhante é utilizado no TCP, entre o pacote de dados e o pacote de reconhecimento (Ack - *Acknowledge*). Os pacotes de interesse têm o mesmo papel

do anúncio da janela deslizante do TCP, ou seja, o tamanho da janela do transmissor pode variar a partir da quantidade de pacotes de interesse enviados (JACOBSON, et al., 2009). Esta regra básica garante que o controle de fluxo seja mantido e permite a comunicação de forma eficiente entre várias máquinas sobre diversas redes heterogêneas.

A segurança nas CCNs é nativa do conteúdo, ou seja, a proteção e a confiança "viajam" com o próprio conteúdo, evitando assim, várias das vulnerabilidades das redes IP atuais. Ao contrário do NetInf, nas CCNs os nomes são tipicamente hierárquicos. No contexto das redes centradas na informação, os mecanismos para resolução de nomes de forma hierárquica podem criar nomes legíveis a humanos, além de fornecerem "dicas" para a resolução de localizadores.

Ainda segundo Jacobson (JACOBSON, et al., 2009), nas CCNs a relação de confiança com o publicador é feita através do prefixo do nome do conteúdo, já que cada pacote enviado contém a assinatura do publicador explícita no seu nome. A assinatura é feita em cada pacote através do seu respectivo nome e estas assinaturas são padrões de chaves públicas fornecidas por uma PKI – *Public Key Infrastructure*. A verificação do nome com o conteúdo é feita através da respectiva chave privada, pertencente ao requisitante do conteúdo. A confiança na chave de assinatura e a integridade dos dados devem ser fornecidas através de mecanismos adicionais. Por exemplo, através de um certificado baseado em alguma PKI.

O paradigma publica/assina (*publish/subscribe*) nas CCNs se dá quando um nó anuncia o nome de determinado conteúdo ao roteador. É assim que a informação é publicada. A assinatura é feita quando um nó interessado no conteúdo envia um pacote de interesse em busca de um potencial publicador. Os pacotes de interesse são enviados por *broadcast* e a escolha de um dado publicador é feita a partir da análise do maior prefixo correspondente ao nome especificado no pacote de interesse. Os pacotes de dados seguem o caminho reverso até alcançar o emitente do pacote de interesse (nó assinante).

2.6.1 Esquema de nomeação CCN

Os nomes CCN são estruturados hierarquicamente e são compostos por três componentes. E ainda, os nomes hierárquicos CCN facilitam a localização e o compartilhamento de dados, uma vez que estes nomes são agregáveis, ou seja, o acesso a estes nomes é mais fácil devido à sua estrutura com semântica hierárquica (JACOBSON, et al., 2009). A **Figura 14** ilustra a estrutura do nome CCN, da qual o primeiro componente corresponde à parte mais a esquerda do nome ou a raiz da estrutura hierárquica, a qual representa o nome global do publicador do conteúdo. O segundo componente representa o nome do conteúdo publicado, descrito na forma organizacional da hierarquia na qual este conteúdo está inserido. O terceiro componente é utilizado para controle de versão e segmentação do conteúdo.

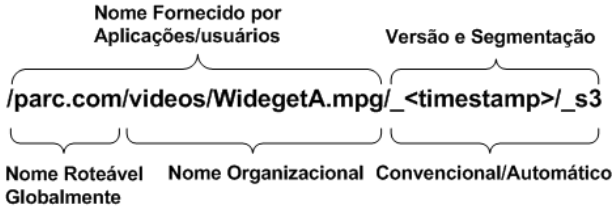

Figura 14. *Composição dos nomes CCN (JACOBSON, et al., 2009).*

O conjunto dos dois primeiros componentes forma um nome legível, com significado próprio, fornecido e utilizado por aplicações e usuários. Existe ainda, uma codificação binária do nome completo do conteúdo, que pode ser criptografado para efeitos de privacidade. A **Figura 15** ilustra a formação do nome de um conteúdo em forma de árvore associada à Figura 14.

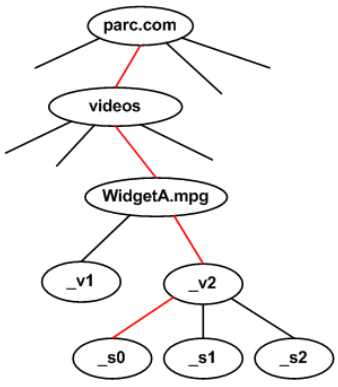

Figura 15. *Ilustração hierárquica de nomes CCN (JACOBSON, et al., 2009).*

2.6.2 Suporte à mobilidade nas CCNs

Dado que as CCNs endereçam conteúdo e não *hosts*, não existe a necessidade de mapear um endereço de camada 3 (endereço IP) a uma identificação de camada 2, como um endereço MAC (*Media Access Control*), por exemplo. Mesmo que haja uma mudança rápida de localização de um *host* devido a um evento de mobilidade, as CCNs podem sempre trocar informações logo que exista uma nova conexão disponível (JACOBSON, et al., 2009).

2.6.3 Roteamento CCN

Como citado anteriormente, nas CCNs o pacote de interesse é roteado para uma potencial fonte que contenha os dados desejados e o pacote de dados retorna pelo mesmo caminho feito pelo pacote de interesse, no sentido reverso até alcançar o assinante do conteúdo. Uma vez que diversos publicadores possuem o conteúdo solicitado pelo assinante, o publicador escolhido será o que possuir o maior prefixo correspondente ao nome especificado no pacote de interesse, daí mais uma peculiaridade dos nomes hierárquicos nesta abordagem.

Segundo Van Jacobson (JACOBSON, et al., 2009), o encaminhamento IP e o encaminhamento CCN são quase idênticos, e as CCNs, portanto, podem utilizar os protocolos de roteamento convencionais, já que elas são compatíveis com o IP, ou podem ser usadas de forma incremental utilizando de forma virtual a infra-estrutura existente.

2.7 PSIRP – Publish/Subscribe Internetworking Routing Paradigm

O PSIRP (AIN, et. al., 2008) é um projeto Europeu financiado pelo FP7 que começou em janeiro de 2008 e tem previsão de término em 2010. O paradigma de roteamento publica/assina (*publish/subscribe*) visa a troca do atual modelo de comunicação da Internet "receptor aceita tudo que lhe for enviado" por um paradigma de comunicação consensual ou autorizada e controlada totalmente pelo receptor (ou assinante). O PSIRP propõe o *projeto* e análise de protocolos criptográficos focados apenas em publicar e assinar o conteúdo, em vez do paradigma enviar e receber da Internet atual. Publicar significa tornar um conjunto de

dados disponível na rede, enquanto a assinatura de um conteúdo corresponde a uma porção de dados solicitada por algum interessado na informação publicada.

Para Wong (WONG, et al., 2008), o paradigma de roteamento publica/assina (*publish/subscribe*) é um mecanismo atraente para a localização e recuperação de conteúdo, uma vez que tal mecanismo trata de forma independente os publicadores e os assinantes de conteúdo. No PSIRP, as mensagens não são dirigidas a qualquer nome de receptor, já que não existem nomes de receptor ou transmissor fornecidos por redes. Desta forma, o roteamento da requisição e do próprio conteúdo no PSIRP é dividido em dois processos: um para o encontro do solicitante (assinante) com uma potencial fonte (publicadora) do conteúdo, e outro destinado à entrega dos pacotes de conteúdo ao assinante.

Quando um nó publica algum dado, a transferência da informação propriamente dita não ocorre. O que ocorre é a publicação da disponibilidade desta informação em um sistema chamado de *rendezvous* ou ponto de encontro. O *rendezvous* é responsável pelo encontro entre o publicador e o assinante de determinada informação. Em outras palavras, quando um nó se inscreve para determinada informação, a rede encontra a publicação de interesse e cria o caminho de entrega entre o publicador e o assinante desta informação.

A arquitetura RTFM (*Rendezvous, Topology, Forwarding and Mediation*) do PSIRP é composta por quatro funções (ESTEVE, et al., 2008 & ZAHEMSZKY, et al., 2009): ponto de encontro (*rendezvous*), gerenciamento de topologia, encaminhamento e mediação. O *rendezvous* é responsável por fazer a ligação de um interessado em um conteúdo com uma ou mais potenciais fontes para este conteúdo. O gerenciamento de topologia constrói e mantém caminhos entre diferentes redes de transmissão baseadas no paradigma publica/assina (*publish/subscribe*). O encaminhamento realiza a entrega de pacotes baseada em técnicas de comutação por rótulos. E finalmente, a mediação refere-se à transmissão física de dados entre dois nós.

Conforme Ain (AIN, et. al., 2008), a arquitetura do PSIRP possui ainda um mecanismo de escopo, utilizado para limitar o alcance de uma informação publicada. Com base em políticas de governança, um usuário pode construir uma relação de confiança e privacidade entre as entidades participantes na comunicação (publicadores, assinantes e a

própria informação). Uma determinada publicação pode estar disponível em diversos escopos e pode ser inserida em um escopo a partir do interesse de um publicador ou de um assinante. A **Figura 16** ilustra o conceito de escopo, onde uma publicação (por exemplo, uma imagem) é disponibilizada para a família e amigos, enquanto a publicação de um *e-mail* pode ser disponibilizada para colegas, por exemplo.

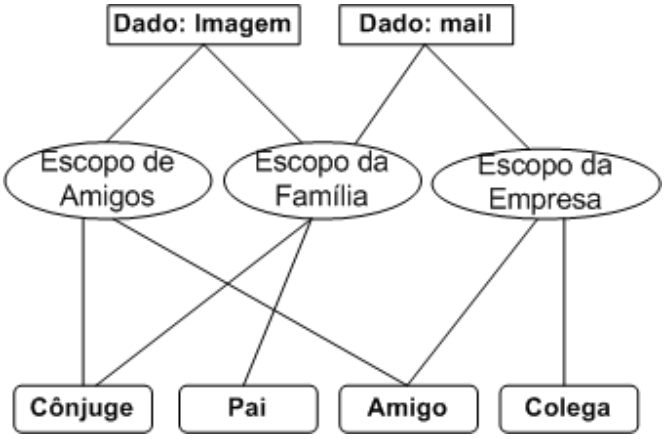

Figura 16. *Representação de escopos PSIRP* (*AIN, et al., 2008*).

Na arquitetura PSIRP existem identificadores que definem as relações entre os objetos de informação nos diferentes níveis do sistema, formando uma arquitetura em camadas. Os sistemas de resolução de identificadores são responsáveis pela interação entre os identificadores situados em diferentes camadas (AIN, et. al., 2008). A **Figura 17** ilustra o sistema de resolução de identificadores em camadas.

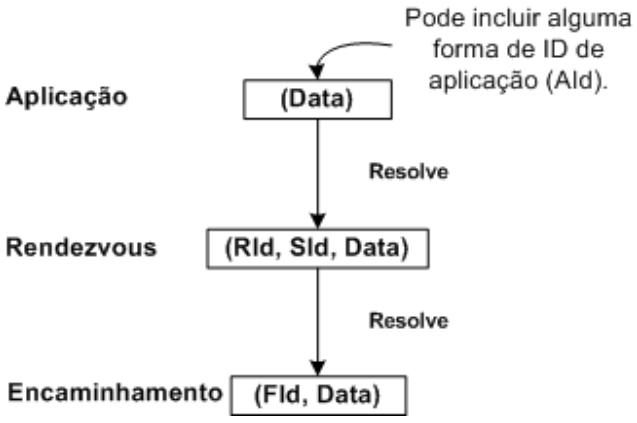

Figura 17. *Sistema de resolução de identificadores do PSIRP* (*AIN, et al., 2008*).

Segundo Ain (AIN, et. al., 2008), os identificadores de aplicação (AIds – *Application Identifiers*) são utilizados por publicadores e assinantes para classificar, distinguir e rastrear entidades como dispositivos, produtos, pessoas, títulos de conteúdos (músicas, endereços de páginas *Web*), serviços (endereço de Serviços *Web* ou de *email*, número de telefone), etc. Estes identificadores, diferente dos outros identificadores a serem apresentados, são definidos usando a semântica da aplicação e contextos específicos.

Os identificadores de *Rendezvous* (RIds – *Rendezvous Identifiers*) são utilizados para ligar os identificadores de nível superior (identificadores de aplicações) com os identificadores das camadas inferiores (identificadores de encaminhamento). Os RIds podem ainda incluir operações como autenticação de rede, controle de acesso de publicadores e de assinantes interessados em participar de um determinado encontro, além de permitir o encaminhamento de solicitações de conteúdo para um determinado publicador (AIN, et. al., 2008).

Os identificadores de encontro podem ser criados por um emissor ou um receptor de dados a fim de estabelecer uma conexão. Em outras palavras, o publicador pode adquirir um identificador e enviar para o assinante, ou um assinante pode selecionar um identificador para o encontro e informar a um potencial publicador. Existe ainda uma terceira opção para a escolha de identificadores, onde uma aplicação ou um serviço necessita de um identificador de encontro para estabelecer a comunicação, por exemplo, em um serviço de *Third Party Call*[3], onde esta terceira parte envolvida adquire um identificador de forma dinâmica e o compartilha com um publicador e um assinante da informação.

Os identificadores de escopo (SIds – *Scope Identifiers*) são utilizados para delimitar a acessibilidade do conteúdo publicado. Os SIds são implementados pelos RIds para limitar a distribuição de conteúdo a regiões específicas e definir a rota para a entrega dos pacotes de dados, ao definir o FId (*Forwarding Identifiers*). Um dos desafios do PSIRP é gerenciar a escalabilidade relacionada às estruturas de escopos, já que para estabelecer um determinado ponto de encontro, precisa-se dimensionar corretamente o escopo relacionado. A

[3] *Third Party Call*, ou chamada de terceiro, significa a requisição a uma rede de telecomunicações (ex: rede de operadora de telefonia) com o objetivo de solicitar o estabelecimento de uma conexão telefônica entre dois terminais fixos e/ou móveis para estabelecer um determinado serviço (CARVALHO, 2008).

delimitação do escopo, portanto, pode ser vista como um equivalente lógico para o conceito de topologias de rede na arquitetura IP.

Ainda segundo Ain (AIN, et. al., 2008), um par (RId, SId) é usado pelo sistema de encontro para determinar o conjunto apropriado de identificadores de encaminhamento para a entrega dos dados para um conjunto de assinantes. Um único RId pode ser associado a um conjunto de SIds. Em outras palavras, uma publicação de dados pode ser associada a um ponto de encontro estabelecido pelo publicador, e ao mesmo tempo, estar ligada a diferentes escopos de acessibilidade. Resumidamente, existe pelo menos um ponto de encontro por escopo. O publicador de uma informação deve publicá-la dentro de um escopo específico, identificando-o durante a operação de publicação.

Os identificadores de encaminhamento (Fids) são utilizados para o transporte das publicações através das redes. Cada par ativo (RId, SId) possui um mapeamento de pelo menos um identificador de encaminhamento. As árvores de transmissão identificadas por um FId podem ser compartilhadas para finalidades de escalabilidade (AIN, et al., 2008).

Para questões de segurança, ainda segundo Ain et al. (AIN, et al., 2008) é esperado que os identificadores sejam concebidos de maneira semelhante à utilizada pelo HIP, ou seja, utilizando o identificador baseado em uma chave pública cuja a correspondente chave privada é de posse do publicador. Isso permite que o *host* publicador escolha um identificador que possa ser usado por assinantes para autenticar os dados e verificar se eles foram enviados de fato pelo publicador que possui a associada chave privada.

Em PSIRP, as propriedades de segurança para mensagens de controle mais importantes são a proteção da integridade e a autenticação. A proteção da integridade pode verificar, por exemplo, se o pacote foi modificado, e a autenticação garante que o publicador possui a permissão do escopo para publicar uma informação com um determinado par (RId, SId) (ZAHEMSZKY, et al., 2009).

Conforme (ZAHEMSZKY, et al., 2009), o PSIRP possui ainda uma autenticação a nível de pacote (PLA - *Packet Level Authentication*). A PLA é uma função para prover segurança aos *hosts* finais através de uma assinatura criptográfica feita individualmente em

cada pacote. Ela permite ainda, que os nós ao longo do caminho de entrega verifiquem os pacotes sem que existam associações de confiança pré-estabelecidas com o publicador. Em PSIRP, a PLA é utilizada geralmente para segurança de mensagens de controle (mensagens publica/assina), podendo ser ampliada para a sua utilização em todo o tráfego.

2.7.1 Suporte à mobilidade PSIRP

Em PSIRP, o suporte à mobilidade pode ser realizado em diferentes níveis na pilha de protocolos e para diferentes tipos de nós finais (AIN, et. al., 2008). O suporte inclui a mobilidade de *hosts* publicadores e assinantes e a mobilidade da rede. Para tanto, mecanismos de *handoff* precisam ser suportados na rede. Isto requer que o roteamento e a topologia de encaminhamento sejam atualizados durante e depois dos *handoffs*.

Para a rede lidar com os *handoffs,* os publicadores precisam restabelecer suas publicações na nova localização e os assinantes, por sua vez, precisam restabelecer as suas assinaturas. Um processo semelhante deve ser feito para o suporte à mobilidade de roteadores e redes móveis. Entretanto, para este caso é necessário restabelecer um conjunto de agregações de publicações e assinaturas. Além disso, um roteador móvel pode desempenhar o papel de um servidor de *rendezvous* para os nós que ele serve, ou atualizar as requisições de publicações e assinaturas junto ao sistema de *rendezvous* responsável pelo seu escopo. Este sistema de *rendezvous,* por sua vez, atualiza as tabelas de roteamento da rede conforme as atualizações de localização.

2.7.2 Roteamento PSIRP

No PSIRP, o roteamento é dividido em dois níveis: um para a resolução de nomes e para o *rendezvous* e outro para o transporte da informação. Resumidamente, o processo de roteamento se desenvolve na localização e na recuperação do conteúdo e pode ser descrito da seguinte forma: primeiramente, para a inicialização de qualquer comunicação, um publicador e um assinante de determinada informação estabelecem o seu respectivo ponto de encontro (*rendezvous*). Posteriormente, o sistema de *rendezvous*, tendo o conhecimento prévio de onde a informação está disponível (no nó do publicador e/ou em *caches* na rede), envia o par de

informações *rendezvous*/escopo para a função de roteamento a fim de determinar uma ou mais árvores de entrega apropriadas, baseando-se na informação da topologia, e finalmente os dados são enviados ao assinante.

Esta distinção entre o *rendezvous* e o roteamento tem uma estrutura semelhante à separação dos nomes de *host* no DNS e os endereços IP (NIKANDER, & MARIAS, 2009). A **Figura 18** ilustra os principais componentes da arquitetura PSIRP.

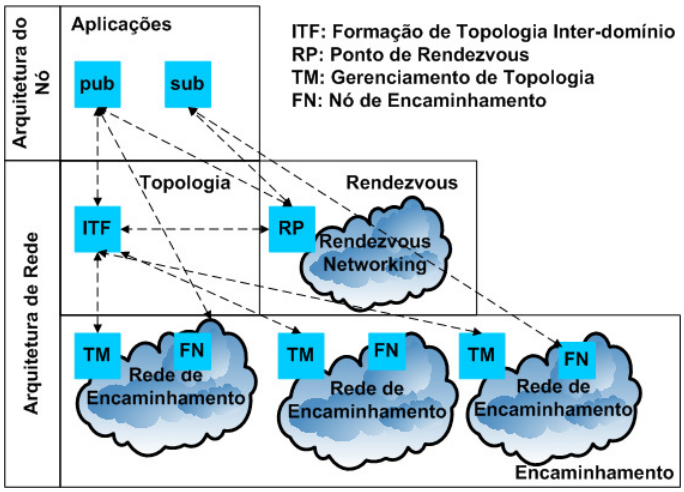

Figura 18. *Arquitetura conceitual PSIRP* (*ZAHEMSZKY, et al., 2009*).

As funções *Pub* e *Sub* são utilizadas por aplicações desenvolvidas nos *hosts.* Elas possibilitam publicações e assinaturas de objetos de informação junto ao sistema de *rendezvous.* Estas publicações e assinaturas são delimitadas através de um determinado escopo (identificado por um Sid). Dado que o ponto de encontro fez a ligação entre um publicador e um assinante, o caminho para a entrega de dados é criado através dos nós de encaminhamento (FN – *Forwarding Nodes*) em negociação com uma função de formação de topologia inter-domínio (ITF – *Inter-domain Topology Formation*), com base nas informações prestadas pelo gerenciamento de topologia (TM – *Topology Management*) (ZAHEMSZKY, et al., 2009).

O PSIRP possui ainda um mecanismo anexado aos pacotes para melhorar o desempenho do encaminhamento de dados, denominado de filtros de Bloom (AHLGREN, et al., 2010). Os filtros de Bloom são uma simples estrutura de dados aleatória com o propósito de consultar e selecionar elementos de forma eficiente em um dado conjunto de participantes, como por exemplo, a escolha de uma rota em um conjunto de rotas. A desvantagem dos filtros

de Bloom é que estes permitem falsos positivos, em contrapartida, a economia de espaço muitas vezes compensa essa desvantagem quando a probabilidade de um erro é controlada ou é suportada por alguma aplicação (AHLGREN, et al., 2010) (PASQUINI, et al., 2010a).

Conforme (BRODER & MITZENMACHER, 2004), os filtros de Bloom têm sido utilizados em aplicações de banco de dados desde 1970, mas só nos últimos anos eles têm se tornados populares na literatura de redes de comunicação. Algumas das aplicações em redes de comunicação auxiliadas pelos filtros de Bloom são: resumo de conteúdo para auxiliar colaborações em sobreposição às redes *peer-to-peer*; recursos de roteamento, algoritmos probabilísticos para localização de recursos; roteamento de pacotes, aceleração e simplificação dos protocolos de roteamento de pacotes; medição de tráfego em roteadores e em outras redes de dispositivos.

2.8 Comparações entre NetInf, CCN e PSIRP

Nesta seção são comparadas e discutidas algumas características e opções de *projeto* relevantes entre as arquiteturas de redes centradas na informação apresentadas.

2.8.1 Nomeação da Informação

Segundo Ahlgren (AHLGREN, et al., 2010), nomes persistentes são um pré-requisito para a localização e disseminação da informação de forma eficiente. Caso contrário, é difícil de se beneficiar das cópias da informação espalhadas em *caches* (até mesmo de cópias armazenadas no próprio local da assinatura do objeto de informação), uma vez que os nomes utilizados são diferentes (não-persistentes). Outro problema conhecido é o erro “404 – Arquivo não encontrado” devido à mobilidade do objeto de informação não utilizando a persistência de nomes.

Os nomes das CCNs são concebidos de forma hierárquica, através da composição de um nome roteável globalmente, um nome organizacional e uma parte convencional e automática para controle de versão e segmentação. A estrutura desses nomes pode ser representada em forma de árvore, sendo que a raiz da estrutura é a parte roteável globalmente e representa o nome do publicador ou o local de armazenamento. Desta forma, a localização

do objeto de informação interfere semanticamente na criação dos nomes para esses objetos, não considerando, portanto, a persistência de nomes e o desacoplamento da identificação e localização da informação.

Já os nomes NetInf e PSIRP são criados de forma persistente, desacoplando, portanto, a identificação e a localização da informação. Estes nomes são semelhantes por usarem um esquema de nomeação plano, e divergem pelo fato do NetInf utilizar um único *namespace*, enquanto o PSIRP utiliza dois *namespaces*, um para o encontro e outro para o encaminhamento. O *namespace* especificado pelo NetInf é semelhante ao do PSIRP destinado ao encontro. Por outro lado, o NetInf não possui o *namespace* destinado ao encaminhamento pois este *namespace* é parte do protocolo particular da camada de transporte (AHLGREN, et al., 2010 & AIN, et al., 2008).

Os esquemas de nomeação hierárquicos têm a vantagem de estabelecer nomes legíveis a humanos. Os nomes planos também podem ser legíveis a humanos. Entretanto, estes nomes podem coincidir e gerar conflitos quando inseridos em um contexto global, dado que eles não possuem uma entidade central que os gerencia de forma a evitar tais conflitos.

Outra vantagem dos sistemas hierárquicos é que estes permitem a agregação de estado de roteamento (*routing state*[4]) além do fato de que o conteúdo de nome hierárquico pode ser criado dinamicamente em resposta a uma solicitação, tal qual acontece nas CCNs. Esta peculiaridade das CCNs só é possível devido à sua estrutura de nomes hierárquicos não-opacos concebidos de forma significativa, isto é, os nomes possuem significado próprio.

Em contrapartida, os nomes planos possuem melhores propriedades de segurança, já que não dependem de entidades centralizadas de resolução de nomes para assegurar esta propriedade, permitindo a auto-identificação das informações. Esta última peculiaridade dos nomes planos é bastante atraente às aplicações que necessitam gerar informações com identificadores de forma autônoma. A questão mais importante, portanto, seria como fazer a ligação entre o esquema de nomeação de forma plana e um sistema de resolução de nomes para endereços como o DNS atual, já que este último é baseado em identificadores

[4] Otimização de roteamento baseada na agregação hierárquica de endereçamento a fim de reduzir o tamanho e a taxa de crescimento do sistema de roteamento global (FULLER & LI, 2006).

hierárquicos e, portanto, inapropriado para identificadores planos. Ou ainda, no caso de soluções *clean-slate*, como relacionar os nomes planos a um localizador e como mapear identificadores em localizadores de forma dinâmica.

Existem na literatura outras propostas de encaminhamento e consulta a sistemas de *cache* baseado em nomes planos como o SPSwitch[5] baseado em filtros de Bloom (ESTEVE, et al., 2008), a consulta para roteamento de conteúdo baseado em DHT (AHLGREN, et al., 2008) (LIU, et al., 2009) e o roteamento baseado em rótulos planos (ROFL) (CAMPISTA, et al., 2010).

2.8.2 Questões de Segurança

O fato de que nas redes centradas no conteúdo as mensagens enviadas contêm controles relacionados apenas ao conteúdo, cria um isolamento natural entre informação e os *hosts* da rede, tornando estes menos vulneráveis, já que usuários mal-intencionados terão mais dificuldade em acessar estes *host*s usando apenas informação fraudulenta. No contexto das redes centradas na informação, as abordagens baseadas em nomes planos estão fortemente ligadas às relações de confiança (*trust*), já que os nomes possuem ligações com o conteúdo ou com o proprietário do conteúdo através de algoritmos criptográficos.

A integridade dos objetos de dados do NetInf é feita através do campo *autenticador*, que faz a ligação direta com o publicador dos dados. “Se o campo de rótulo contiver um valor de *hash* criptográfico calculado sobre o objeto de dados, o nome também conterá uma ligação direta com o conteúdo, que fornece, assim, a auto-certificação da integridade do objeto de dados” (AHLGREN, et al., 2010).

[5] O SPSwitch tem como proposta considerar o problema genérico de tomar decisões de encaminhamento de alta velocidade (Gbps), com base em um vasto universo de identificadores planos (valores *hash* criptográficos de 256 *bits*, por exemplo) (ESTEVE, et al., 2008).

Nas CCNs a relação com o publicador é feita através do seu prefixo único explícito em cada pacote de dados enviado. A assinatura é feita em cada pacote através do respectivo nome e estas assinaturas são padrões de chave pública. A verificação do nome com o conteúdo é feita através da respectiva chave privada. A confiança na chave de assinatura e a integridade dos dados devem ser fornecidas através de mecanismos adicionais. Por exemplo, um certificado baseado em PKI.

Para o PSIRP a segurança é garantida através dos identificadores de encontro e escopo construídos a partir do *hash* do conteúdo, além de possuir um mecanismo de segurança indireto através do *hash* de chave pública adicionado ao rótulo. Mais ainda, o PSIRP dispõe da autenticação em nível de pacote (PLA) adicionada para o encaminhamento de pacotes a fim de combater ataques DoS (AHLGREN, et al., 2010).

2.8.3 Roteamento

Nas CCNs o pacote de interesse é roteado por *broadcast* para uma potencial fonte que contenha os dados desejados. O publicador escolhido será o que possuir o maior prefixo correspondente ao nome especificado no pacote de interesse, daí mais uma peculiaridade dos nomes hierárquicos nesta abordagem. Os pacotes de dados seguem o caminho inverso até alcançar o solicitante. Além disso, segundo Jacobson (JACOBSON, et al., 2009), as CCNs podem utilizar os protocolos de roteamento convencionais, já que elas são compatíveis com as redes IP.

O roteamento tanto no NetInf como no PSIRP é dividido em dois processos: um para o encontro do solicitante com uma potencial fonte, e outro destinado à entrega dos pacotes propriamente dita. O PSIRP possui ainda um mecanismo de encaminhamento de dados baseados em filtros de Bloom anexado aos pacotes. Já o NetInf possui o mecanismo de roteamento baseado em nomes (LLC), que integra os caminhos de resolução e recuperação, e que segundo Ahlgren (AHLGREN, et al., 2008) pode resultar em melhor desempenho.

2.8.4 Aspectos de *Caching*

Um dos pré-requisitos das redes centradas na informação é que os usuários podem se beneficiar de cópias de conteúdos espalhados ao longo da rede, permitindo otimizar a disseminação geral da informação. O *cache* de objetos de informação, portanto, é parte crucial das abordagens de redes centradas na informação.

Na arquitetura NetInf, o conteúdo em *cache* também pode ser encontrado através de busca local ou ser encontrado através do sistema de resolução de nomes se, neste caso, existir uma cópia em *cache* neste local. Há também a possibilidade do transporte NetInf encontrar cópias em *cache* no caminho para a localização de uma cópia do objeto. Já no PSIRP, o *cache* de conteúdo se limita ao escopo e ao ponto de encontro. Nas CCNs, os conteúdos em *cache* podem ser localizados através de um mecanismo de pesquisa local ou ao longo do caminho do pacote de interesse até a potencial fonte (AHLGREN, et al., 2010).

2.8.5 Relação entre o publicador e o assinante

Como citado anteriormente, nas CCNs, apenas quando um nó anuncia um prefixo de determinado conteúdo ao roteador a informação é publicada. A assinatura é feita quando um nó interessado no conteúdo envia um pacote de interesse em busca de um potencial publicador. Entretanto, nas CCNs o publicador pode criar o conteúdo dinamicamente a partir de um pacote de interesse enviado pelo assinante. Em outras palavras, um interessado pode construir um nome para determinado conteúdo que ainda não exista e o publicador desenvolve um objeto de informação em resposta, completando o encontro.

A publicação no NetInf é feita ao registrar o nome de determinado objeto de informação em um sistema de resolução de nomes e a assinatura é feita a partir da solicitação do objeto publicado ao sistema de resolução de nomes. Contudo, o ponto de encontro entre o emissor e o receptor NetInf é estabelecido apenas depois que os identificadores são registrados em um sistema de resolução de nomes. Já no PSIRP, o contato entre o publicador e o assinante é feito através do processo de *rendezvous*.

2.8.6 Escalabilidade

A escalabilidade é um pré-requisito para qualquer projeto de redes de nova geração. Os projetos devem estar preparados para crescer e alcançar um escopo global, permitindo que trilhões de nós e terminais sejam conectados à rede, sendo capazes de transportar os *exabytes* de informação mensal da Internet atual e futura, além de possibilitar a integração de novas funcionalidades, sem comprometer a base da arquitetura principal e ainda ser economicamente viável (AHLGREN, et al., 2010) (MINTS, 2010).

Conforme Ahlgren (AHLGREN, et al., 2010), os roteadores CCNs encontram dois desafios ao lidarem com a escala: o gerenciamento do número de prefixos de nomes e o armazenamento de informações de estado por pacote (*Per-Packet State*). As informações de estado são necessárias ao longo do caminho fim a fim, o que representa uma desvantagem do ponto de vista da escalabilidade.

Ainda segundo Ahlgren (AHLGREN, et al., 2010), um desafio do PSIRP é gerenciar a escalabilidade relacionada às estruturas de escopos, já que para estabelecer um determinado ponto de encontro, precisa-se dimensionar corretamente o escopo relacionado. Outro desafio é o gerenciamento do encaminhamento baseado no filtro de Bloom, que se torna um problema quando usado com grandes grupos de destinatários. O problema da escalabilidade enfrentado pela arquitetura NetInf é que cada objeto de informação deve ser representado por uma entrada no sistema de resolução de nomes global.

2.8.7 Outros aspectos

As abordagens de redes centradas no conteúdo devem suportar a mobilidade tanto dos objetos de informação, quanto de *hosts* e redes. Para o NetInf e o PSIRP, a mobilidade da informação é garantida pela persistência de nomes e pelo desacoplamento entre a identificação e a localização da informação. Enquanto a mobilidade de entidades físicas (nós e redes) é suportada por mecanismos adicionais de desacoplamento da identificação e localização de *hosts*, semelhantes ao *Mobile* IP (acrescentando o *care-of-address*) e ao HIP (conceito de *rendezvous*), para o NetInf e PSIRP, respectivamente.

Dado que a localização do objeto de informação influencia semanticamente nos nomes CCNs, estas redes não desacoplam a identificação e a localização da informação. O desacoplamento não acontece também para entidades físicas, como *hosts* e redes. Segundo Jacobson (JACOBSON, et al., 2009), as CCNs endereçam conteúdo e não *hosts*, não existe a necessidade de mapear um endereço a partir de um identificador. Neste caso, uma conexão interrompida devido a um evento de mobilidade pode ser restabelecida logo que exista uma conexão disponível.

A **Tabela 3** é uma versão ampliada da tabela do documento número 6.2 de (AHLGREN, et al., 2010) e resume as comparações feitas entre as abordagens de redes centradas no conteúdo.

Tabela 3. *Algumas comparações entre as redes centradas no conteúdo (AHLGREN, et al., 2010).*

	NetInf	**CCN**	**PSIRP**
Esquema de Nomeação	Um *namespace* plano.	Nomes hierárquicos.	Dois *namespaces* planos. Um para o *Rendezvous*, outro para o transporte.
Nomes	Nomes persistentes, opacos e não agregáveis.	Nomes não persistentes, legíveis e agregáveis.	Nomes persistentes, opacos e não agregáveis.
Segurança	Ligação direta do nome com a chave pública.	Ligação com o publicador através do prefixo do nome.	*Hash* de conteúdo + *hash* chave pública + PLA.
Roteamento	Dois roteamentos: um para encontro, outro para transporte; LLC.	*Broadcast* do pacote de interesse até a fonte; os dados seguem o caminho reverso.	Dois roteamentos: Um para encontro, outro para transporte + filtro de Bloom.
Caching	Localizado pelo mecanismo de resolução de nomes, de pesquisa local ou no transporte.	Localizado por pesquisa local ou no caminho para o publicador.	Limitado ao escopo do ponto de encontro.
Relação entre o Publicador e o Assinante	Pub: Objeto de Informação As: Consulta resolução de nomes; O encontro necessita do registro prévio do nome com o NRS.	Pub: Prefixo de nome As: Envia o interesse O encontro pode ser criado dinamicamente em resposta a um interesse.	Pub: Estabelece o ponto de encontro As: Estabelece contato; O encontro é desenvolvido no *Rendezvous.*
Escalabilidade	Uma entrada no NRS por objeto; agregação de publicação.	Relacionada ao Nº de prefixos; estado de encaminhamento por pacote.	Relacionada às estruturas dos escopos.
Desacoplamento ID/LOC *Hosts*	Semelhante ao *Mobile* IP – *Home Address e Care-of Address.*	Mapeamento ID/LOC considerado desnecessário.	Semelhante ao HIP - *Rendezvous.*
Desacoplamento ID/LOC Informação	Desacopla – Persistência de nomes.	Não desacopla – Localização influencia semanticamente no nome.	Desacopla – Persistência de nomes.

A principal diferença entre as três abordagens é o sistema de nomeação, já que os demais desafios como roteamento, segurança e escalabilidade estão fortemente ligados ao esquema de nomeação escolhido. A arquitetura NetInf e o PSIRP utilizam o esquema de resolução de nomes na forma plana e a CCN utiliza o esquema de nomeação de forma hierárquica. Para as CCNs, os nomes da informação e dos *hosts* são criados com base na localização. Desta forma, as CCNs não desacoplam a identificação e a localização destas entidades. Apesar dos sistemas planos possuírem uma ligação mais próxima com os requisitos de segurança, estes sistemas sofrem com a incompatibilidade com o DNS atual e, portanto,

necessitam de um mecanismo que interopere com o DNS ou a concepção de um sistema de resolução de nomes interligado a um sistema de roteamento baseado em nomes planos que seja escalável.

2.9 Conclusão

As arquiteturas de desacoplamento entre a identificação e a localização de *hosts* formam uma proposta para o suporte à mobilidade e *multihoming* de *hosts*, dentre outros problemas decorrentes do acoplamento lógico existente no endereço IP. Contudo, a principal diferença entre as arquiteturas propostas estão na forma como os identificadores são concebidos e na flexibilidade da utilização de tais arquiteturas em ambientes heterogêneos.

Já as redes centradas na informação propõem o desenvolvimento de uma rede com o foco na informação propriamente dita em vez de *hosts*, se preocupando, sobretudo, com a segurança, escalabilidade, eficiência, qualidade e suporte a comunicação em tempo real. Tal abordagem pode ser classificada como *clean slate*, uma vez que redesenha do zero a forma como trocamos informação na Internet. Estas redes têm como principal função a interligação em grande escala dos desenvolvedores e consumidores de conteúdo aumentando a eficiência no acesso e na disseminação da informação.

O grande desafio das redes centradas na informação é estabelecer um padrão para nomeação da informação de forma consistente e eficiente, além de soluções para localização e roteamento das informações baseadas nas informações nomeadas. Além disso, os identificadores precisam ser persistentes de forma a suportar o desacoplamento entre a identificação e a localização das entidades. Segundo Ahlgren (AHLGREN, et al., 2010), a escolha da forma como os nomes são criados e gerenciados pode ser a parte mais importante da arquitetura.

Capítulo 3

Indireções e DHTs

Este capítulo aborda o conceito de indireções e a relação entre indireções e as arquiteturas de redes citadas no Capítulo 2. Neste contexto, é levantada uma forma na qual tais indireções podem ser tratadas através de mapeamentos utilizando a abordagem das DHTs. Para tanto, é explicado e exemplificado o funcionamento da tecnologia DHT e alguns benefícios propostos pela utilização de tal tecnologia são levantados. Além disto, propõe-se também uma forma na qual os mapeamentos de indireções podem ser armazenados e localizados. Alguns exemplos de algoritmos DHTs são usados para ilustrar a proposta.

3.1 Indireções

O acréscimo de funcionalidades como a mobilidade, *multihoming*, *multicast* e *anycast* quebra o paradigma ponto a ponto da Internet original, que define uma localização única e fixa para os *hosts* conectados a uma rede. Entretanto, as indireções formam uma abstração para ajudar na resolução das limitações da arquitetura original a fim de suportar o acréscimo destas novas funcionalidades.

As soluções baseadas em indireção assumem um ponto de indireção físico ou lógico colocado entre o transmissor e o receptor para encaminhar o tráfego entre eles. Ao se comunicar através de um ponto de indireção, e não diretamente entre os *hosts*, o transmissor pode abstrair a localização do receptor para suportar a mobilidade, ou o número de receptores para suportar o *multicast*. Para exemplificar, o *Mobile* IP insere o *care-of-address* para abstrair

o localizador do *host* e suportar a mobilidade, enquanto o *Multicast* IP assume um ponto lógico de indireção para abstrair o número de receptores e seus localizadores (STOICA, et al., 2002).

Os sistemas de indireções podem ser usados, por exemplo, para referenciar uma entidade através de um nome, um rótulo ou um local onde esta entidade está armazenada. Neste caso, uma forma comum de aplicar uma indireção é manipular um valor através do seu endereço lógico ou utilizar o conceito de apontador para determinada posição de memória (INDIRECTION, 2010). Outro exemplo de sistema de indireção é o DNS, que é responsável por prover um mecanismo de nomes de tal forma que estes nomes possam ser mapeados para um endereço IP.

No contexto dos protocolos de desacoplamento de identificação e localização, um exemplo do conceito de indireção é aplicado ao fazer o mapeamento entre o identificador e o localizador dos *hosts*. Para o protocolo HIP, esta indireção é desenvolvida por uma nova camada de identificação de *host*, responsável por fazer uma ligação dinâmica entre o identificador do *host* e o seu localizador (endereço IP). Pontos de indireção acontecem de forma semelhante no protocolo de desacoplamento do projeto Akari e no protocolo MILSA na camada de identidade e na subcamada de mapeamento, respectivamente. Já o *Mobile* IP acrescenta um endereço utilizado para suporte à mobilidade, denominado de *care-of-address*. Este novo endereço corresponde ao ponto de indireção neste protocolo. No LISP, o ponto de indireção é aplicado ao tunelar pacotes entre o ITR e o ETR.

As arquiteturas NetInf e PSIRP aplicam o conceito de indireção ao desacoplar os objetos de informação de seus locais de armazenamento. Neste contexto, é inserida uma forma de indireção para tratar o encontro entre o publicador e o assinante de um conteúdo, como por exemplo, os servidores de *rendezvous* implementados pelo PSIRP. Na arquitetura NetInf, o sistema de resolução de nomes é desenvolvido através de DHTs, as quais representam um ponto de indireção responsável por mapear o identificador em um localizador da informação. Dado que as CCNs não aplicam o conceito de desacoplamento de identificação e localização das entidades, a PIT pode ser considerada um ponto de indireção, já que esta tabela armazena os pacotes de interesses expressos pelos assinantes até que o serviço de encaminhamento entregue estes pacotes a uma potencial fonte publicadora.

3.2 A tecnologia DHT

A proposta de arquitetura para o Sistema Generalizado de Resolução de Indireções a ser especificada no Capítulo 4 utiliza o conceito de DHTs para a formação, manipulação e armazenamento dos mapeamentos entre os atributos (nomes, identificadores, localizadores e descritores) tanto dos objetos de conteúdo, quanto dos *hosts* responsáveis pelo armazenamento destes objetos. Neste contexto, para recuperar um objeto de conteúdo é necessário um identificador deste conteúdo, bem como um localizador. E, resumidamente, o identificador deste conteúdo é fornecido através de uma função *hash* do mesmo e o localizador associado a este identificador é recuperado através de uma consulta DHT.

Conforme (DISTRIBUTED HASH TABLE, 2010) e (LIU, et al., 2009), as DHTs são uma classe de sistemas distribuídos descentralizados que prestam um serviço de pesquisa semelhante a uma tabela *hash*. Pares <*chave, valor*> são armazenados nas DHTs e qualquer nó pode participar recuperando o valor associado a uma dada chave. A responsabilidade de manter o mapeamento entre as chaves e valores é distribuída entre os nós. Este fato permite às DHTs escalarem para um número extremamente grande de nós e lidar com conectividade intermitente de nós na rede.

Os nós são organizados em uma topologia de rede sobreposta (*overlay*) construída usando como base uma arquitetura de rede existente. A topologia virtual depende do algoritmo utilizado para implementar a DHT. Por exemplo, o algoritmo Chord (STOICA, et al., 2003) organiza os nós virtuais em um círculo, enquanto o algoritmo Kademlia (MAYMOUNKOV & MAZIERES, 2002) os organiza como folhas de uma árvore binária. Desta forma, o roteamento de mensagens dentro do ambiente DHT é relacionado com a estrutura da topologia DHT e o algoritmo utilizado para implementá-la.

Para auxílio à escalabilidade das propostas, a responsabilidade pela manutenção das tabelas de mapeamento é distribuída entre os nós participantes. Conforme Cirani e Veltri (CIRANI & VELTRI, 2007), cada nó mantém uma pequena tabela de roteamento contendo o identificador e o localizador (para as redes IP, o endereço IP) de alguns dos seus pares vizinhos. A distribuição destes mapeamentos torna a abordagem DHT eficiente, de forma que os nós não precisam conhecer todos os seus vizinhos, e o alcance a um nó qualquer é

conseguido com um número relativamente pequeno de saltos, variando conforme o algoritmo utilizado.

A **Figura 19** ilustra de forma generalizada o funcionamento de uma DHT.

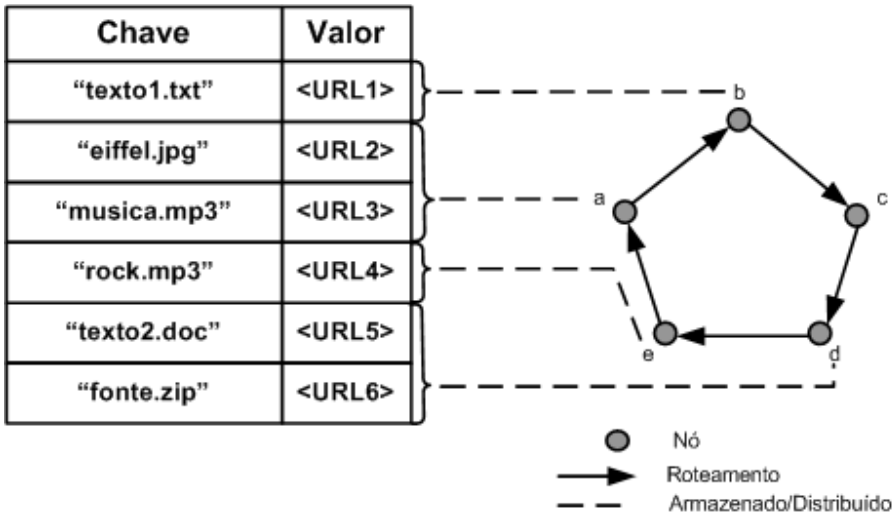

Figura 19. *Mapeamentos de chaves e valores distribuídos entre nós DHTs* (*GHODSI, 2006*).

O mapeamento é feito entre os nomes de arquivos que são utilizados como chaves em consultas DHTs e os respectivos localizadores URL (*Uniform Resource Locator*) destes arquivos. Os registros DHTs contendo chaves e valores são distribuídos entre os nós *a, b, c, d* e *e*, os quais mantêm também informações sobre a identificação e localização de outros nós da topologia virtual. Considere, por exemplo, que uma consulta pelo nome *"eiffel.jpg"* seja feita ao nó *c.* Para encontrar a URL deste objeto de informação é necessário que conjuntos de chaves sejam associadas a cada nó, como será visto na Seção 3.5.

3.3 Chamadas de procedimentos das DHTs

Cada algoritmo DHT define um protocolo, que contém geralmente, um conjunto de chamadas de procedimento remoto (RPCs - *Remote Procedure Calls*) para serem utilizados na comunicação e cooperação entre os nós DHTs. As interfaces de programação de aplicações (APIs) são responsáveis por intermediar as DHTs e suas aplicações através de métodos RPCs. Segundo Cirani e Veltri (CIRANI & VELTRI, 2007) e Lua, et al., (LUA, et al., 2005), as RPCs podem ser definidas abstratamente como:

- *put (key,value):* este método é usado para armazenar um par <*chave,valor*> na DHT;
- *value = get (key):* este método é usado para recuperar as informações armazenadas na DHT que estão associadas a uma dada chave;
- *remove (key):* este método é usado para remover um par <*chave,valor*> na DHT.

Ainda conforme Cirani e Veltri (CIRANI & VELTRI, 2007), a chave de um recurso (serviço, conteúdo ou *host*) geralmente é formada pelo *hashing* do nome do recurso através de uma função *hash* definida pelo algoritmo DHT, enquanto que o valor é um dado associado ao recurso (com alguns metadados e/ou endereço de localização onde o recurso pode ser encontrado). A **Figura 20** reproduzida de (LUA, et al., 2005) ilustra as chamadas de procedimento remoto (RPCs) das DHTs inseridas em um contexto de compartilhamento de arquivos entre pares (*peers*).

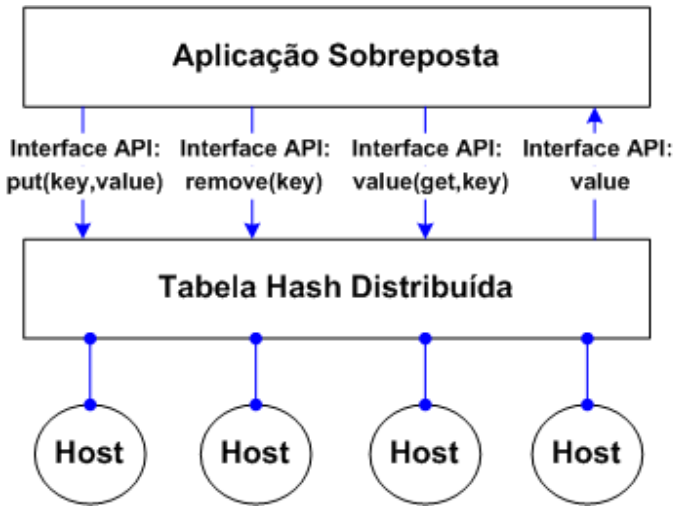

Figura 20. *Chamadas de procedimento remoto em DHTs (LUA, et al., 2005).*

3.4 Características suportadas pelas DHTs

Na Internet atual, toda comunicação entre dois *hosts* depende de um serviço de localização, tal como o DNS, para mapear um URI em um endereço IP que especifica onde o recurso identificado pelo URI pode ser acessado. No entanto, os serviços de localização (LS – *Location Services*) geralmente introduzem pontos centralizados para consultas de localizadores, criando pontos de vulnerabilidade no sistema. Desta forma, devido à natureza

distribuída das DHTs, elas apresentam uma forma perfeita para a criação de serviços de localização distribuídos (DLS – *Distributed Location Services*) (CIRANI & VELTRI, 2007).

As DHTs formam uma infraestrutura que pode ser usada para criar outros serviços complexos, tais como sistemas de arquivos distribuídos, compartilhamento de arquivos *peer-to-peer*, sistemas de distribuição de conteúdo, *web caching* cooperativo, *multicast*, *anycast*, serviços de nome de domínio e mensagens instantâneas (LIU, et al., 2009). Elas possuem propriedades inerentes de auto-gerenciamento, auto-configuração, robustez, tolerância a falhas e alta escalabilidade. Portanto, o uso de DHTs para criar sistemas de mapeamento é desejável e bastante promissor (CIRANI & VELTRI, 2007) e (LIU, et al., 2009).

Ainda sobre as propriedades das DHTs, Hanka et al., (HANKA, et al., 2008) concluíram que estas tabelas formam um conceito eficiente para os requisitos das redes de nova geração. Em especial, elas podem suportar:

- *nomenclatura e endereçamento:* nomes e endereços para *hosts* e informação, com mapeamento dinâmico entre eles;
- *escalabilidade:* roteamento altamente escalável, suportado naturalmente pelas DHTs;
- *descentralização:* eficientes mecanismos de busca, de gerenciamento e de localização de entidades altamente distribuídos, propondo melhorias na disponibilidade e acessibilidade de conteúdos;
- *mobilidade:* através do mapeamento dinâmico entre identificadores e localizadores para *hosts* e informação;
- *auto-organização:* diminui o custo de manutenção e operação, melhora a resiliência e permite a auto-cura devido à natureza distribuída e ciente da localização de nós nas DHTs.

A **Figura 21** reproduzida de (HANKA, et al., 2008) ilustra alguns dos benefícios propostos pelas DHTs.

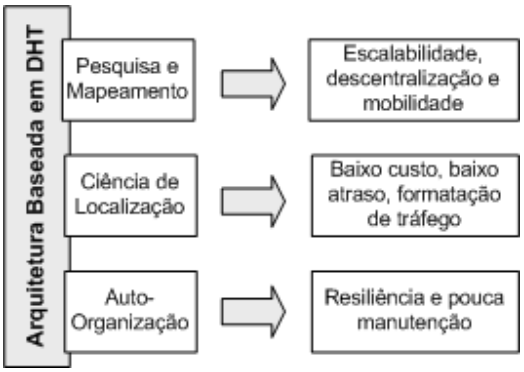

Figura 21. *Resumo de funcionalidades propostas pelas DHTs (HANKA, et al., 2008).*

3.5 Algoritmos DHTs

Existem diversas abordagens diferentes para tratar mapeamentos baseados em DHTs. Apesar de terem o mesmo objetivo, que é distribuir e localizar um conjunto de registros de forma eficiente para um conjunto de nós participantes, os algoritmos DHTs diferem entre si na forma como os nós são organizados e na forma como as consultas aos valores distribuídos entre estes nós são roteadas.

Tais algoritmos são utilizados largamente em redes par a par (P2P – *Peer-to-Peer*). Estas redes são compostas por um grande número de nós, denominado de pares, com o objetivo geral de compartilhar dados e recursos entre os pares. Os nós P2P agem tanto como cliente quanto servidor. Esta característica torna a rede mais robusta, dado que a responsabilidade pelo armazenamento das informações é distribuída entre os pares, tornando a topologia menos vulnerável, uma vez que não existe um ponto centralizado de armazenamento e coordenação (RIECHE, et. al., 2004).

Nas próximas seções será feita uma apresentação e discussão de um apanhado de algoritmos DHTs, com o objetivo de selecionar uma forma na qual os mapeamentos para a resolução de indireções propostos no Capítulo 4 poderão ser armazenados e consultados.

3.5.1 Chord

O protocolo Chord (STOICA, et al., 2003) organiza um conjunto de 2^m nós em um círculo unidirecional, onde *m* é o número de *bits* utilizado para a construção das chaves destes nós. Desta forma, a cada conteúdo e nó é associada uma chave de *m bits*. As chaves de conteúdos são utilizadas como entrada nas consultas DHTs e os valores de retorno informam o localizador do conteúdo ou do nó associado à chave da consulta.

A **Figura 22** ilustra a formação de uma topologia para o algoritmo Chord com um exemplo de distribuição das chaves aos nós.

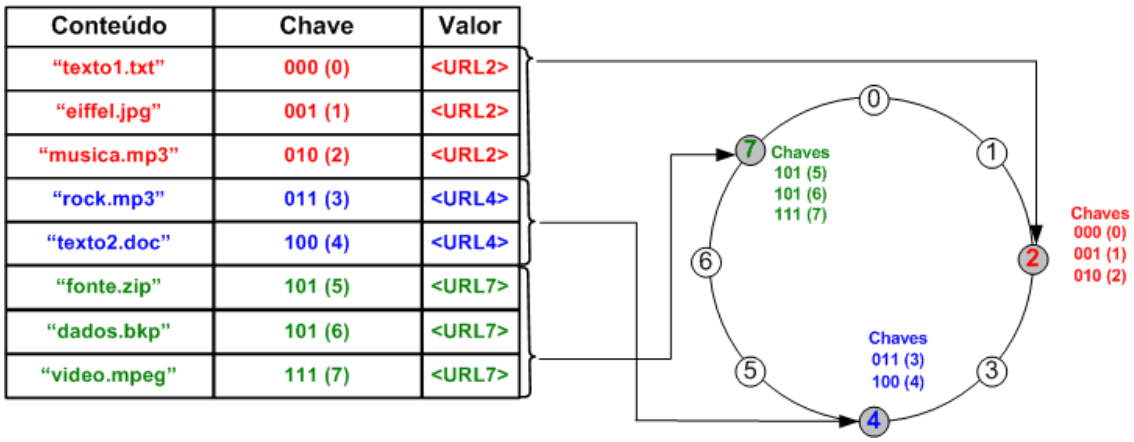

Figura 22. *Exemplo de distribuição de chaves para o algoritmo Chord.*

Considere o cenário ilustrado pela Figura 22. Com *m* igual a 3 *bits*, a topologia virtual consiste, portanto, de 8 (0 a 7) espaços para organizar os nós. Os identificadores dos nós são ordenados em um círculo e cada chave *k* (de um objeto de conteúdo, por exemplo) é associada ao primeiro nó cujo identificador seja igual ou posterior ao identificador de *k* no espaço virtual de endereçamento. Considere ainda que, neste exemplo, estão conectados apenas os nós 2, 4 e 7 (círculos em cinza). Desta forma, as chaves *k* igual a 0, 1 e 2 são associadas ao nó 2, assim como as chaves 3 e 4 são associadas ao nó 4 e as chaves 5, 6 e 7 são associadas ao nó 7.

Para este mesmo exemplo, considere que um usuário ou aplicação requisite ao nó 4 o localizador do arquivo *eiffel.jpg*. Dado que o nó 4 não possui o localizador do arquivo requisitado, ele encaminha a requisição para o nó 7, que por sua vez encaminha a requisição para o nó 2, que pode responder à requisição, dado que ele conhece o localizador do arquivo *eiffel.jpg*.

No algoritmo Chord, os nós conhecem apenas o seu sucessor (2 conhece apenas 4, 4 conhece 7, etc.). Considerando o pior caso, uma requisição pode percorrer toda a topologia (N nós) para encontrar o localizador de uma dada chave, o que torna tal abordagem ineficiente. Assim, para melhorar a eficiência deste processo, o protocolo Chord mantém informações adicionais de roteamento para outros nós além do seu sucessor em uma tabela denominada de *finger table.*

As *finger tables* possuem informações sobre outras chaves distribuídas sobre a topologia virtual da rede, bem como informações sobre o nó sucessor responsável pelo armazenamento de um intervalo de chaves. Segundo (STOICA, et al., 2003), cada nó mantém uma *finger table* com no máximo *m* (quantidade de *bits* para a composição das chaves) entradas correspondentes aos seus respectivos nós, escolhidos de forma eficiente. A chave correspondente a i-ésima entrada na *finger table* em cada nó *n* contém o identificador do primeiro nó *s*, que sucede *n* por pelo menos 2^{i-1}, onde *i* varia de 1 até *m*. Ou seja, a i-ésima entrada é igual a $(n+2^{i-1})$ para valores de *i* variando de 1 até *m.*

A **Figura 23**, ilustra a formação das *finger tables* para os nós 2, 4 e 7.

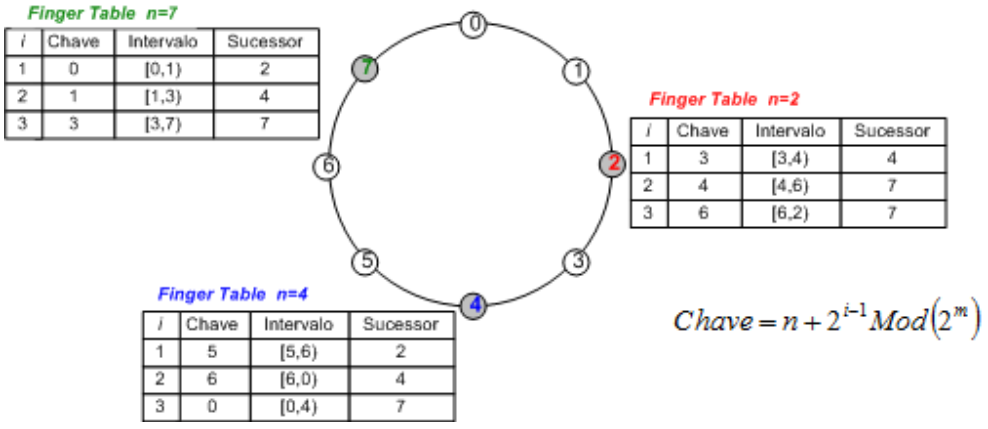

Figura 23. *Exemplo do algoritmo Chord com Finger Tables.*

Voltando ao exemplo anterior, no qual um usuário ou aplicação requisite ao nó 4 o localizador do arquivo *eiffel.jpg* (com identificador igual a 1). Dado que o nó 4 não possui o localizador do arquivo requisitado, agora a *finger table* é consultada sobre o intervalo responsável pelo identificador 1. Neste caso o nó 4 verifica a terceira entrada da sua *finger table* correspondente ao intervalo [0,4), o qual pertence a chave 1, com o nó sucessor igual a 7. Como 7 precede 1, o nó 4 irá requisitar ao nó 7 o sucessor do identificador 1. Neste caso, o nó 7 irá concluir, através da sua tabela, que o sucessor do identificador 1 é o nó 2, retornando o valor do nó 2 à consulta feita pelo nó 4.

Ainda segundo Stoica (STOICA, et al., 2003), se a distância entre o nó que recebe a consulta e seu predecessor (o pior caso) é reduzida ao meio em cada um destes saltos e sendo 2^n a quantidade de nós na topologia virtual, em no máximo *n* saltos uma chave pode ser alcançada. Desta forma, a complexidade é reduzida de O(N) (sem utilizar as *finger tables*) para O(log(N)) saltos, utilizando as *finger tables.* Por exemplo, em uma rede Chord com 1024 nós, as consultas de chaves utilizando as *finger tables* são resolvidas em no máximo O(10) saltos. Além disso, Stoica et. al. (STOICA, et al., 2003) concluíram que, na prática, as requisições de consultas podem ser resolvidas em 0,5(log(N)) saltos.

Note ainda que, para lidar com a conectividade intermitente de nós nesta topologia, os registros armazenados em um nó 4 (da Figura 23) são transferidos para um próximo nó 7, no caso em que o nó 4 se desconecte desta topologia. Da mesma forma, registros acumulados em um nó 4 podem ser transferidos para um nó 5*,* no caso em que este último se conecte na topologia virtual da rede.

3.5.2 Kademlia

O Protocolo Kademlia (MAYMOUNKOV & MAZIERES, 2002) organiza os nós em forma de árvore binária. Cada nó e objeto de conteúdo Kademlia possui um identificador de 160 *bits*. Os registros de objetos de conteúdo são armazenados em nós cujas chaves são "próximas" às chaves dos objetos de conteúdo. Para publicar ou localizar um registro contendo um par <*chave,valor*>, o algoritmo Kademlia baseia-se em uma noção de distância entre as chaves do nó consultado e a chave do objeto de conteúdo de interesse. Ou seja,

quando uma chave *x* (de um conteúdo, por exemplo) é pesquisada em um nó de chave *y*, ambas as chaves são comparadas e a diferença entre elas representa a sua distância.

Para comparar a chave de interesse e a chave do nó no qual a pesquisa da chave de interesse está sendo feita, o Kademlia utiliza a função ou exclusivo (XOR), a qual compara ambas as chaves *bit* por *bit* e retorna a diferença entre elas. Ao contrário do Chord, nos algoritmos que utilizam a função XOR para a descoberta de uma dada chave como o Kademlia e o Pastry (ROWSTRON & DRUSCHEL, 2001), a distância entre duas quaisquer chaves *x* e *y* é simétrica, ou seja, $d(x,y) = d(y,x)$ para todo valor de *x* e *y* (GOTZ, et. al., 2005).

Para o encaminhamento de requisições, cada nó precisa conhecer e manter informações de no mínimo um nó de cada uma das sub-árvores nas quais ele não está contido. A **Figura 24** ilustra uma topologia de rede Kademlia formada com identificadores de 4 *bits*, contendo portanto, 16 nós. As elipses em cinza representam as sub-árvores que devem conhecer e que devem ser conhecidas do nó 4.

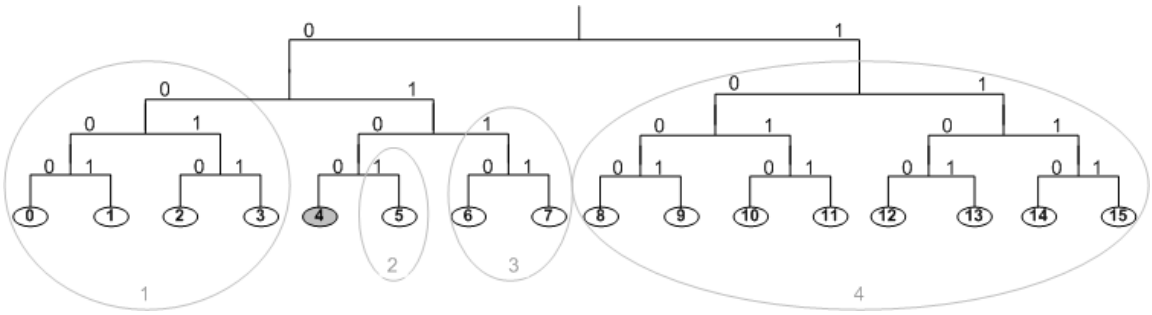

Figura 24. *Exemplo de topologia Kademlia.*

Considerando a Figura 24 e o exemplo dado na seção anterior, se uma aplicação ou usuário requisitar ao nó 4 (com a chave 0100) o localizador do arquivo *eiffel.jpg* de chave 1 (0001), o nó 4 primeiramente verifica que a sub-árvore que contém a chave desejada (baseando-se nos primeiros *bits* da chave) é a *sub-árvore 1,* e encaminha esta requisição para um dos seus conhecidos nós daquela sub-árvore. A escolha entre os nós conhecidos será o que possuir o maior prefixo em comum com a chave requisitada. Neste caso, dado que a requisição foi encaminhada para um nó da *sub-árvore 1*, este verifica se possui ou não a chave requisitada. Caso não possuir, a requisição volta a ser encaminhada para um terceiro nó até que a chave de destino seja alcançada.

Conforme (GOTZ, et. al., 2005), dado que cada nó conhece pelo menos um nó em cada sub-árvore, uma consulta de uma chave arbitrária pode ser resolvida em O(log(N)) saltos de roteamento, em média. Além disso, em vários casos, um nó pode conhecer mais de um nó nas outras sub-árvores. Semelhante ao protocolo Pastry (ROWSTRON & DRUSCHEL, 2001), o algoritmo Kademlia sugere o encaminhamento de consultas para vários nós em paralelo. Esta abordagem, apesar de aumentar o uso da largura de banda disponível, diminui a latência das consultas e aumenta a robustez da topologia, aumentando a eficiência das consultas e tornando a topologia mais tolerante a falhas.

3.5.3 Pastry

No protocolo Pastry (ROWSTRON & DRUSCHEL, 2001), os identificadores dos nós e dos objetos de conteúdo são compostos por 128 *bits*. O Pastry é semelhante ao Chord por organizar os nós em círculo contendo 2^m nós e semelhante ao Kademlia, ao localizar uma determinada chave através da razão de proximidade entre a chave do objeto de conteúdo consultado e a chave do nó que conhece este objeto de conteúdo.

Neste protocolo, cópias de conteúdo são armazenadas em nós *cache* com identificadores próximos numericamente aos identificadores destes conteúdos. Quando um usuário ou aplicação requisita a chave de um conteúdo a um determinado nó, se este nó não conhecer a chave, a requisição é encaminhada para outro nó que possui o identificador mais próximo da chave consultada. O número esperado de saltos para responder a consulta é de O(log(N)), onde N é o número de nós na topologia virtual Pastry.

Segundo (GOTZ, et. al., 2005), todo nó Pastry possui uma tabela de roteamento (*routing table*) com informações sobre outros nós, através da qual a chave pesquisada é comparada com o identificador dos nós conhecidos. A escolha do nó a ser encaminhada a requisição será o que possuir o identificador mais próximo à chave consultada. A *routing table* atinge um objetivo semelhante à *finger table* do protocolo Chord. Ou seja, ambas mantêm informações sobre um número de nós distribuídos ao longo da topologia da rede. Uma segunda tabela, denominada de *leaf set* contém informações sobre um conjunto de nós com identificadores mais próximos ao identificador do nó atual. As informações prestadas pelo *leaf*

set são semelhantes às informações prestadas pelo campo *nós sucessores* contida na *finger table* do protocolo Chord.

A relação de proximidade entre duas chaves é medida pelo maior número de dígitos (contando da esquerda para a direita) em comum entre elas. Entretanto, esta proximidade é uma relação lógica e não indica uma distância geográfica ou uma distância em saltos de roteamento em uma dada topologia de rede.

Com base neste fato, o protocolo Pastry mantém uma terceira tabela com informações de distância geográfica denominada de *neighborhood set*. Esta tabela mede a distância entre duas chaves com base em uma métrica de proximidade de rede escalar, assumida como já informada pela infra-estrutura de rede existente, como por exemplo, os passos de roteamento IP para a localização topológica dos nós. Vale ressaltar que as informações contidas nesta tabela não são utilizadas para escolher um nó para o qual a requisição vai ser encaminhada (a escolha é feita comparado-se as chaves), e sim para aumentar a eficiência do encaminhamento da requisição, através da escolha de um melhor caminho entre dois nós.

A **Figura 25** ilustra as informações contidas nas tabelas para um nó arbitrário com identificador 103220.

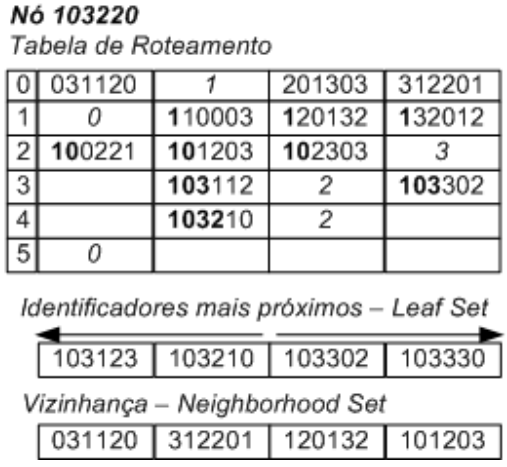

Figura 25. *Exemplo de formação de tabelas para o protocolo Pastry (GOTZ, et al., 2005).*

Para o exemplo da Figura 25 (GOTZ, et. al., 2005), os valores utilizados são: *m=12 bits* (chave do identificador)*; N=4096* (nós na topologia); *b=2* (parâmetro de configuração da base de identificação 2^b, geralmente igual a 4 no algoritmo Pastry, formando uma base hexadecimal, portanto). A quantidade de linhas na tabela de roteamento é igual a $\log_{2^b} N$, com 2^b-1 entradas em cada linha.

A tabela de roteamento para este exemplo contém 6 linhas e 3 entradas por linha. O índice de linhas da tabela de roteamento (primeira coluna) corresponde à quantidade "*n"* de dígitos semelhantes entre o identificador do nó atual e os identificadores relacionados em cada linha (os dígitos em comum de cada identificador estão em negrito). O dígito "*n+1"* está dentro das possibilidades 2^b-1 dígitos diferentes do atual e está representado na tabela por números em itálico.

Caso nenhum nó conhecido possua um identificador pertinente a uma célula da tabela, esta célula é mantida vazia. Já o *leaf set* informa os nós com identificadores mais próximos do nó 103220, que não representa, necessariamente, o nó mais próximo geograficamente. Enquanto, o *neighborhood set* informa os nós mais próximos (com métricas escalares) ao nó 103220 nesta topologia virtual.

Ainda segundo (GOTZ, et. al., 2005), genericamente, uma chave é consultada primeiramente no *leaf set*. Se a chave consultada estiver contida neste conjunto, a consulta é então encaminhada diretamente para o nó destinatário. Caso contrário, se o *leaf set* não possuir a chave consultada, a chave é então pesquisada na *routing table*. Se esta tabela também não possuir a chave consultada, a consulta é encaminhada para outro nó que possua um prefixo em comum com a chave pesquisada e que esteja numericamente mais próximo da chave do que o nó atual.

3.6 Conclusão

Em síntese, as indireções estão presentes tanto nas propostas para a separação entre identificadores e localizadores de *hosts*¸ quanto para as arquiteturas NetInf e PSIRP, ao desacoplar a identificação e a localização dos objetos de informação. Neste contexto, a tecnologia DHT mostra-se bastante atraente para criar sistemas de mapeamentos devido à sua

característica altamente escalável e às suas propriedades inerentes de auto-gerenciamento, auto-configuração, robustez e tolerância a falhas.

Desta forma, no Capítulo 4 será proposto um Sistema Generalizado de Resolução de Indireções para tratar diversos mapeamentos entre as entidades, propondo uma solução para alguns dos desafios delineados no Capítulo 2, baseando-se na abordagem DHT para o armazenamento, localização e roteamento de consultas a estes mapeamentos.

Capítulo 4

Sistema Generalizado de Resolução de Indireções

Neste capítulo apresenta-se um Sistema Generalizado de Resolução de Indireções (GIRS - *Generalized Indirection Resolution System*) para tratar indireções utilizadas em propostas de Internet do Futuro, em especial nas redes centradas na informação. Estas indireções são resolvidas através de mapeamentos de atributos (nomes, identificadores, localizadores e descritores) tanto de objetos de conteúdo, quanto dos *hosts* responsáveis pelo armazenamento destes objetos. Além disso, a abordagem para tratar tais mapeamentos é focada em um ponto de convergência para a resolução de indireções de forma generalizada e holística.

Um GIRS pode ser projetado considerando que: (i) todas as entidades (reais e virtuais) possuem um padrão único (*Padrão*) que as identifica; (ii) para facilitar o uso por parte das pessoas, a todas as entidades podemos dar um nome legível (*Nome*), em linguagem natural; (iii) a todas entidades de rede unicamente identificadas podemos associar um objeto que as descreve (*Descritor*); (iv) as entidades possuem um localizador que pode representar a sua localização na topologia da rede (*Localizador*); (v) os atributos (*padrão, nome, descritor e localizador*) de uma mesma entidade ou de entidades distintas se relacionam através de mapeamentos para a localização e recuperação de objetos de conteúdo.

Desta forma, um mesmo sistema de indireção pode ser usado para mapear atributos tanto da informação, quanto de *hosts*. Ele também pode ser usado para tratar atributos de outras entidades, tais como pessoas, serviços, *hardware*, aplicações, terminais, fluxos de informações, redes, etc. A **Figura 26** ilustra a estrutura do GIRS.

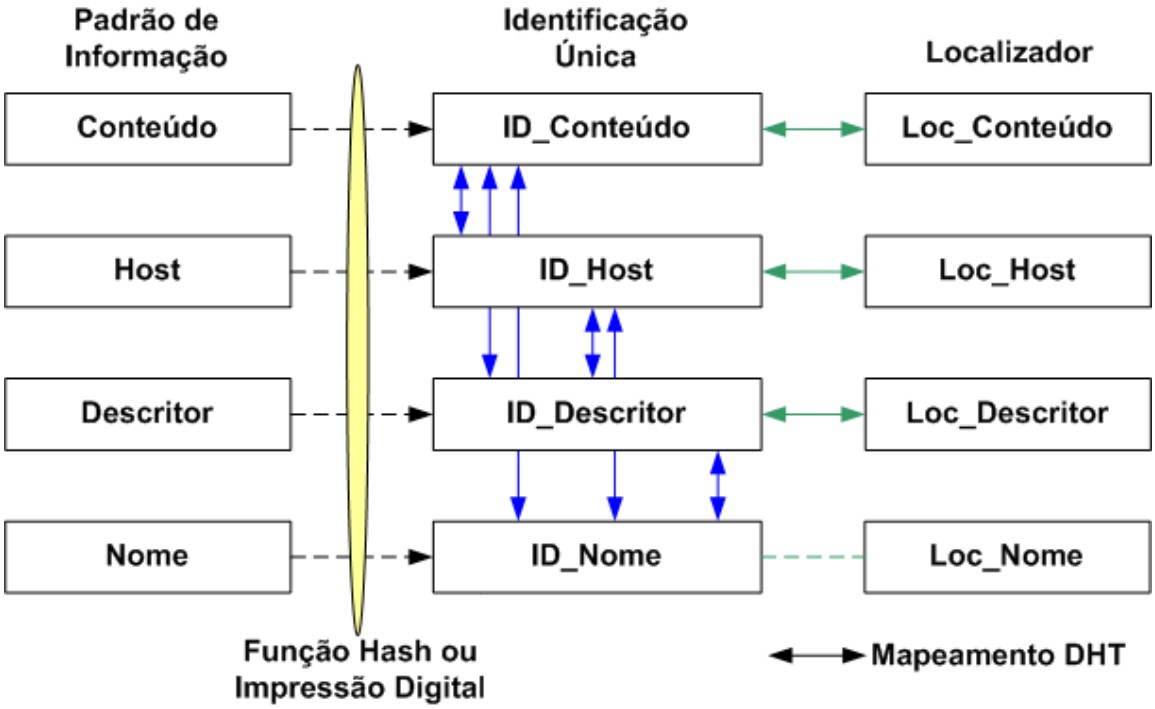

Figura 26. *Estrutura do GIRS.*

Considere um objeto de conteúdo digital, tal como uma foto. O padrão de *bits* desta foto é estatisticamente único. Ou seja, a possibilidade de que outra foto que não seja uma cópia ter este mesmo padrão é estatisticamente desprezível. Portanto, podemos usar este padrão para criar uma identificação também única para este objeto. Para cada entidade identificada unicamente, podemos associar um descritor (detalhado posteriormente na seção 4.3), que a descreve para uso geral. Podemos ainda associar um mesmo descritor a mais de uma identidade, no caso em que existam variações de formato. Por fim, podemos associar um nome a este objeto.

Observamos que esta mesma seqüencia de indireções pode ser usada para *hosts*. Cada *host* tem um padrão único, que neste caso pode ser obtido a partir de uma combinação única de características da máquina. Isto já acontece para autenticar acessos a banco no Brasil. A partir do padrão estatisticamente único, é possível se criar um identificador único para este *host*. Neste caso, a relação entre identificação e descrição pode ser de um para um. Por fim, podemos dar um nome legível a este *host*, tal qual fazemos hoje.

Observe que este sistema permite relacionar descritores de entidades com identificadores de outras. Na Figura 26, por exemplo, ao invés de relacionarmos um objeto de conteúdo diretamente com um localizador de *host*, tal qual é feito pelas propostas de redes centradas na informação abordadas no Capítulo 2, podemos relacionar com um identificador de *host*. Desta forma, o localizador do *host* é obtido diretamente e dinamicamente através do seu identificador, tal qual acontece no protocolo HIP, fazendo com que o GIRS suporte naturalmente a mobilidade de entidades. Vale a pena ressaltar que o identificador de *host* pode ser associado a vários localizadores, fazendo com que a proposta suporte naturalmente o *multihoming* do *host*. E mais, como a descrição de uma entidade pode conter mais de um identificador de *host*/servidor/etc, isto torna possível o roteamento *anycast*, bem como o roteamento via fontes e trajetos múltiplos (de várias fontes, tal como *peer-to-peer*).

Outro aspecto é que a nomenclatura é uniformizada para todas as entidades, podendo ser em linguagem natural. O sistema permite determinar o relacionamento entre identificadores de diversas entidades, criando uma cadeia de rastreabilidade e de contexto baseada em nomes.

4.1 Dicionário e classificação dos mapeamentos

Os atributos (nomes, identificadores, localizadores e descritores) tanto de objetos de conteúdo, quanto de *hosts* são relacionados através de mapeamentos que representam registros em DHTs. O dicionário, por sua vez, é uma estrutura de dados responsável por manter um conjunto de mapeamentos entre as chaves utilizadas como entrada em consultas DHTs e seus respectivos valores de saída. Ele pode ser usado por usuários e aplicações para resolução de valores relacionados a uma dada chave como, por exemplo, a resolução de um

localizador de *host* associado a um identificador (do próprio *host* ou de um objeto de conteúdo), ou a resolução de um identificador de descritor a partir de um identificador de conteúdo.

A **Figura 27** é uma versão modificada de (OHLMAN, et al., 2009) e ilustra a estrutura do mapeamento para o GIRS.

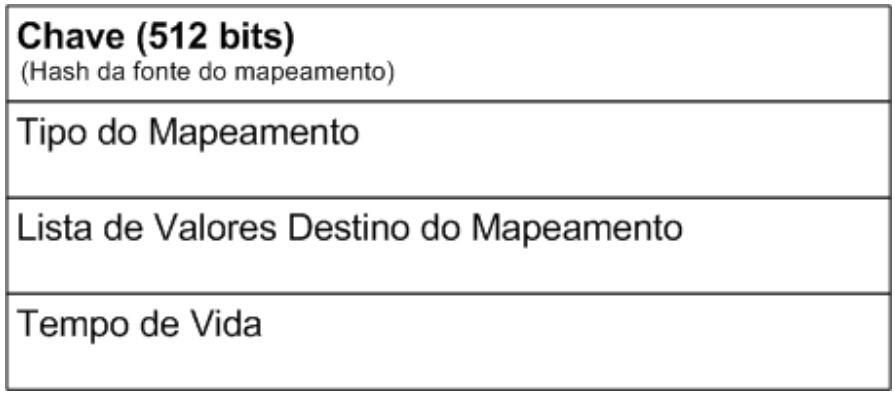

Figura 27. *Estrutura dos mapeamentos para o GIRS.*

Genericamente, um mapeamento corresponde a ligação de um identificador a um conjunto de valores associados a este identificador, ou seja, um mapeamento do tipo ID:{Tipo de Mapeamento; Lista de Valores; Tempo de Vida}, onde:

- *Chave (ID):* representa o valor de entrada nas pesquisas DHTs. As chaves são obtidas a partir da função *hash* de 512 *bits* do objeto fonte do mapeamento;
- *Tipo de Mapeamento:* indica a classe do mapeamento e o estilo de resolução entre as entidades;
- *Lista de Valores:* corresponde aos valores retornados pelas resoluções associados a um dado ID. Podem ser únicos ou uma lista de IDs, um ou mais localizadores, etc;
- *Tempo de Vida:* para políticas de atualização e de expiração dos mapeamentos.

Assim como ilustrado na Figura 26, os identificadores das entidades lógicas são obtidos pelo *hashing* do padrão de *bits* destas entidades, enquanto os identificadores de *hosts* são gerados a partir da "impressão digital" dos mesmos. Estes identificadores são utilizados

como entrada (chave) para as requisições de resolução. Desta forma, estas chaves são consideradas estatisticamente únicas, assim como as entidades utilizadas como chave de consultas possuem características ou padrões que as tornam únicas.

Dado que existem vários tipos de mapeamentos, com diversos tipos de chaves e valores de retorno, estes mapeamentos precisam ser classificados para receberem tratamentos específicos. Em outras palavras, a classe do mapeamento informa o tipo da chave de entrada e o tipo do valor de retorno em uma dada consulta. Esta classificação direciona a consulta para uma resolução específica, evitando, portanto, que o mapeamento retorne valores diferentes de valores da classe esperada. Para exemplificar, o identificador de um conteúdo, utilizado como chave de uma consulta pode requisitar como resposta o localizador do conteúdo, ou o identificador do *host* publicador deste conteúdo. Assim, o que diferencia estes dois mapeamentos são as suas classes.

Além disso, um mesmo mapeamento pode ser utilizado tanto no sentido direto, quanto no sentido inverso, sendo classificado, portanto, em classes diferentes. Neste caso, para o exemplo anterior, tem-se o mapeamento de um identificador de *host* para um identificador ou uma lista de identificadores de conteúdo. Desta forma, esta proposta de arquitetura de resolução de indireções utiliza um campo *tipo* com um tamanho de 10 *bits* para a classificação dos mapeamentos e a cada tipo de mapeamento, atribui-se um tipo específico de DHT para a resolução de tal mapeamento.

A **Tabela 4** contém um exemplo para algumas classes de mapeamentos com as suas respectivas chaves de entrada, quantidade e valores de saída, além do tempo de vida (TTL – *Time to Live*) destas classes.

Tabela 4. *Classes de mapeamentos para o Sistema Generalizado de Resolução de Indireções.*

Mapeamento	Chave	Valor (XML)			
		Tipo	Quant	Lista de Valores	TTL
<ID_Nome,ID_Descritor>	ID_Nome	1	253	ID_Descritor1,..., ID_Descritor253. e.g. Lista de descritores de objetos associados ao ID do nome “Torre Eiffel”.	1
<ID_Descritor,ID_Nome>	ID_Descritor	2	3	ID_Nome1,..., ID_Nome3. e.g. IDs de nomes associado a um determinado descritor, tal como “Torre Eiffel”, “Paris”, “Atrações Paris”.	1
<ID_Descritor,ID_Conteúdo>	ID_Descritor	3	4	ID_Conteudo1,...,ID_Conteudo4. e.g. Lista de conteúdos associados a um dado descritor relativo ao nome “Torre Eiffel”.	1
<ID_Conteúdo,ID_Descritor>	ID_Conteúdo	4	1	ID_Descritor1. e.g. ID de descritor associado a um dado conteúdo, tal como uma imagem única eiffel.jpg.	1
<ID_Conteúdo,ID_Host>	ID_Conteúdo	5	7	ID_Host1,...,ID_Host7 e.g. Listta de identificadores de hosts onde se encontram cópias do conteúdo requisitado.	1
<ID_Host,ID_Conteúdo>	ID_Host	6	27	ID_Conteúdo1,..., Conteúdo27. e.g. IDs de conteúdos armazenados em um dado *host*.	1
<ID_Nome,ID_Conteúdo>	ID_Nome	7	253	ID_Conteúdo1,..., Conteúdo253. e.g. IDs de conteúdos que tem exatamente o mesmo nome.	1
<ID_Conteúdo,ID_Nome>	ID_Host	8	3	ID_Nome1,..., ID_Nome3. e.g. IDs de nomes associados a um único conteúdo.	1
<ID_Descritor,ID_Host>	ID_Descritor	9	2	ID_Host1,..., Host2. e.g. Lista de ID de hosts que contém o descritor de um determinado conteúdo.	1
<ID_Host,ID_Descritor>	ID_Host	10	4	ID_Descritor1....,ID_Descritor4. e.g. Lista de descritores que estão disponíveis em um *host* identificado por ID_Host.	1
<ID_Nome,ID_Host>	ID_Nome	11	1	ID_Host1. e.g. Retorna o identificador de um *host* identificado pelo ID_Nome.	1
<ID_Host,ID_Nome>	ID_Host	12	1	ID_Nome1. e.g. Retorna o identificador de um nome associado unicamente a um host com ID_Host.	1
<ID_Descritor,Loc_Descritor>	ID_Descritor	13	2	Loc1_Descritor,...,Loc2_Descritor. e.g. Lista de localizadores do descritor identificado por ID_Descriptor.	1
<ID_Conteúdo,Loc_Conteúdo>	ID_Conteúdo	14	2	Loc1_Conteúdo,...,Loc2_Conteúdo. e.g. Lista de localização do conteúdo identificado por ID_Conteúdo.	1
<ID_Host,Loc_Host>	ID_Host	15	1	Loc1_Host. e.g. Endereço do host na rede.	1

A Tabela 4 relaciona 15 tipos de mapeamentos, os quais representam ligações dinâmicas entre atributos dos *hosts* e de objetos de informação. A coluna da tabela denominada de *Mapeamento* descreve o tipo de valor de entrada (*Chave*) e tipo dos valores de saída relacionados na coluna *Lista de Valores.* Já a coluna *Tipo, Quant.* e *TTL* representam a classe do mapeamento, a quantidade de registros atuais relacionados à chave consultada e o tempo de vida das classes, respectivamente.

Para exemplificar, o mapeamento do tipo 3 é utilizado para consultar identificadores de conteúdo (*ID_Conteúdo*) relacionados a uma chave *ID_Descritor*. Para este exemplo, o mapeamento retorna 4 valores (*ID_Conteúdo1, ..., ID_Conteúdo4*) relacionados à chave requisitada. Já o mapeamento do tipo 5 retorna 7 identificadores de *hosts*, os quais armazenam o conteúdo relacionado à chave consultada, enquanto o mapeamento 14 retorna o localizador (neste caso, localizador único) do *host* associado ao *ID_Host* consultado.

Note que as classes de mapeamentos listadas na Tabela 4 são relacionadas ao GIRS ilustrado pela Figura 26. Apesar do escopo deste trabalho se limitar às classes relacionadas nesta tabela (em especial às classes utilizadas para a localização e recuperação de objetos de informação, definidas posteriormente, no estudo de caso apresentado na Seção 4.3), esta abordagem é totalmente flexível, permitindo a inserção de novas entidades como usuários, serviços, *hardware*, aplicações, terminais, fluxos de informações, redes, dentre outras, bem como a inserção de novas classes de mapeamento relacionadas a estas novas entidades.

4.2 Mecanismos de resolução

Para reduzir a complexidade da proposta, o mecanismo de resolução de indireções é dividido em três módulos baseados nas classes de mapeamentos: (i) um mecanismo de busca (SE – *Search Engine*), que fornece pesquisa e recuperação de entidades com base na semântica, metadados e atributos destas entidades; (ii) um sistema de resolução de nomes, responsável por mapear identificadores em localizadores de objetos de informação (NRS); (iii) e um mecanismo para mapear identificadores e localizadores de *hosts* (*Split Host*).

Desta forma, a resolução dos mapeamentos listados pela Tabela 4 é dividida de acordo com as suas classes. Além disso, novos sistemas de resolução podem ser implementados para suprir a demanda de novos tipos de mapeamentos. Apesar de existir a separação entre os mecanismos de resolução, um mesmo sistema poderia desempenhar as funções dos três módulos, resolvendo de forma generalizada todos os tipos de mapeamentos listados na Tabela 4, bem como novos mapeamentos relacionados a novas entidades.

4.3 Estrutura do descritor

Um descritor é um objeto que contém características do padrão que ele descreve. Tais características são organizadas através de um conjunto de atributos, permitindo melhorar a eficiência de pesquisas, dado que os atributos provêm informações contextuais do padrão em questão.

Genericamente, os atributos podem representar: um nome do objeto descrito como *Eiffel Tower,* Fotos de Paris ou Beethoven; informações de classificação do padrão descrito como imagem, som, serviço, vídeo, página *web*; formato do arquivo como JPEG ou MP3; taxa de *bits*, *codec* utilizado, além de atributos do mundo real como posição GPS ou código de barras.

Note que não existe um padrão pré-estabelecido para os atributos dos descritores. Entretanto, tais atributos devem ser descritos em linguagem natural, permitindo que os usuários e aplicações localizem os padrões desejados com base na semântica destes padrões fornecidos pelos descritores. Desta forma, os descritores podem ser desenvolvidos baseados em uma linguagem que forneça informações legíveis, como por exemplo, a XML (*Extensible Markup Language*) ou a OWL (*Web Ontology Language*), dando suporte para a busca, pesquisa e interpretação das entidades através dos descritores.

A **Figura 28** ilustra uma estrutura genérica para os descritores.

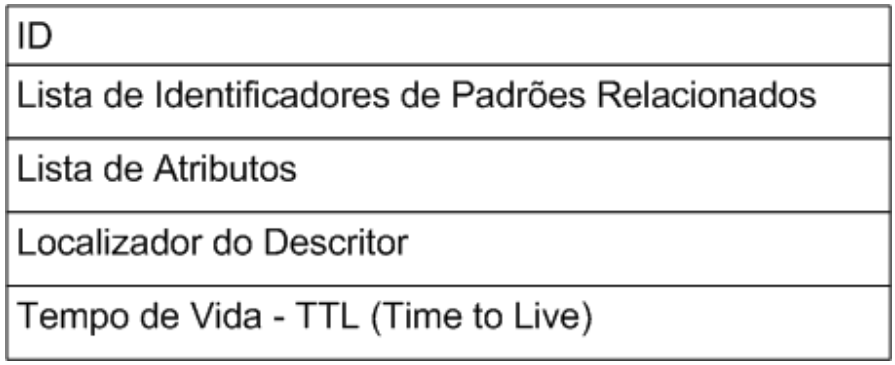

Figura 28. *Estrutura do descritor para o GIRS.*

Todo objeto descritor contém: (i) um identificador (*ID*), gerado através do *hashing* do padrão de *bits* do descritor. Este identificador pode ser usado, por exemplo, como chave de consulta em um mapeamento responsável por retornar o identificador do padrão descrito (mapeamento do tipo 3 definido na Tabela 4); (ii) um identificador ou uma lista de identificadores de objetos de conteúdo ou de *hosts* descritos (*Lista de Identificadores de Padrões Relacionados*); (iii) uma *Lista de Atributos*, responsável por descrever o padrão ou um conjunto de padrões relacionados; (iv) o localizador ou identificador do *host* responsável por armazenar tal objeto descritor (*Localizador do Descritor*); (v) e um *Tempo de Vida* para ser aplicado em políticas de atualização e expiração.

4.4 Codificação de fonte e ajuste de nomes

Os identificadores gerados através de funções *hash* não são legíveis, não possuem informações de semântica e de contexto dos objetos identificados, nem possuem alguma informação hierárquica. Desta forma, a motivação da utilização de nomes nesta proposta é identificar legivelmente as entidades (*hosts* e objetos de conteúdo), de forma que estes nomes possam auxiliar o processo de busca e localização dos objetos.

Entretanto, usuários e aplicações podem nomear as entidades de forma arbitrária e esta arbitrariedade pode ocasionar resultados indesejáveis. Em uma escala global, existe grande possibilidade de inúmeras entidades serem homônimas, e este fato é agravado ao permitir que nomes sejam criados com semântica fraca, como por exemplo, vários computadores serem nomeados por *"computador01"* ou inúmeros objetos de conteúdo serem nomeados por *"musica.MP3"*. Este fato acarretaria em um grande problema de escalabilidade, sobretudo para os mecanismos de busca.

Desta forma, deve-se exigir que os nomes de entidades sejam concebidos utilizando alto grau de semântica, como para o exemplo anterior, *"computador01 departamento de compras da empresa ABC"* ou *"9ª sinfonia de Beethoven.MP3"*. Para tratar tais problemas de nomeação, propõe-se que os nomes devam passar por uma codificação de fonte e um ajuste de tamanho mínimo.

A codificação de fonte é utilizada no processo de geração de identificadores para os nomes. Tal codificação é utilizada para a padronização na nomeação das entidades (*hosts* e objetos de conteúdo). O ajuste padroniza o nome em um tamanho mínimo pré-estabelecido, auxiliando o tratamento de nomes arbitrariamente pequenos. Além disso, o processo de codificação de fonte e ajuste de nomes é genérico para utilizar diferentes linguagens e representações de caracteres. Desta forma, pode-se utilizar qualquer simbologia ou alfabeto para nomear as entidades, uma vez que a codificação de fonte ASCII/Unicode (*American Standard Code for Information Interchange*) transforma os nomes em padrões binários.

A **Figura 29** ilustra o processo de criação de nomes compatíveis com nomes arbitrariamente pequenos.

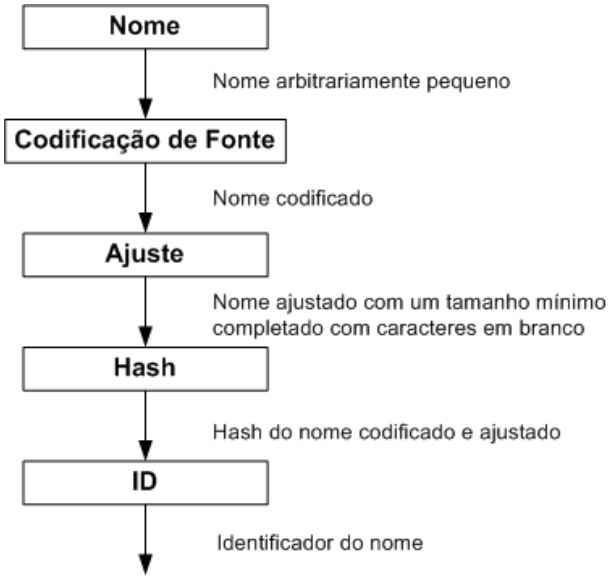

Figura 29. *Codificação de fonte e ajuste para nomes arbitrariamente pequenos.*

4.5 Questões de segurança, privacidade e escopo

Na arquitetura de rede atual, a segurança é baseada em *hosts*, isto significa que é necessária a confiança entre ambas as partes envolvidas em um processo de comunicação. Entretanto, com o acoplamento lógico entre o identificador e o localizador do *host* existente no endereço IP, o identificador de um *host* muda devido a um evento de mobilidade. Esta mudança de identificador forma um ponto de vulnerabilidade nesta arquitetura, dado que dificulta a rastreabilidade do identificador, ao tentar localizar o originador de um ataque, por exemplo.

Já as redes centradas na informação propõem que a segurança e a confiança sejam nativas do próprio conteúdo. Isto quer dizer que um assinante de determinado conteúdo não precisa, necessariamente, conhecer e confiar no publicador deste conteúdo. Tal conteúdo é auto-certificável, permitindo verificar a integridade dos dados independentemente do *host* que enviou o conteúdo ou da confiança no canal de transmissão. Além disso, a autenticação do proprietário abordada no NetInf (DANNEWITZ, 2009) permite tirar conclusões sobre a qualidade do conteúdo e a confiabilidade do seu publicador.

Além disso, as CCNs (JACOBSON, et al., 2009) propõem que cada pacote de dados contenha a assinatura do seu publicador, evitando assim o não repúdio do conteúdo em função do publicador. Enquanto no PSIRP (AIN, et al., 2008), os identificadores do ponto de encontro e do escopo associado a uma publicação é gerado através de uma função *hash* do conteúdo, além de utilizar autenticação em nível de pacotes.

Assim como nas redes centradas na informação, esta proposta de arquitetura utiliza o paradigma publica/assina (*publish/subscribe*), no qual toda troca de informação só pode ser feita através de um consenso entre o publicador e o assinante desta informação. Este fato cria uma barreia de proteção natural para os usuários e máquinas, evitando comunicações indesejáveis como *spams* e, conseqüentemente, fornecendo mais confiança para estes usuários.

O conceito de escopo é utilizado no PSIRP (AIN, et al., 2008) para definir esferas de privacidade. Estes escopos podem ser utilizados para delimitar o âmbito de uma publicação, construindo uma relação de confiança e privacidade entre o publicador e os assinantes de um determinado conteúdo. Ao delimitar um escopo, um usuário pode, por exemplo, definir um grupo de participantes com permissão para acessar uma determinada publicação.

4.6 Escalabilidade

Assim como discutido anteriormente, a escalabilidade é um dos principais desafios para projetar uma rede de nova geração. As arquiteturas de redes do futuro devem ser projetadas para alcançar um escopo global, permitindo que a rede cresça sem comprometer o *projeto* original, além de ser genérica e expansível, possibilitando a integração de novas funcionalidades de acordo com a demanda de novas tecnologias.

Neste contexto, as abordagens baseadas em DHTs mostram-se extremamente eficientes, dado que elas trabalham de forma descentralizada ao distribuir a responsabilidade e a carga do sistema para os nós participantes de uma dada topologia. Para diminuir a complexidade dos mapeamentos DHTs e suportar a escalabilidade, Hanka et al. em (HANKA, et al., 2008) ainda propuseram uma divisão da topologia da rede em uma estrutura de duas camadas.

Com isso, os mapeamentos entre identificadores e localizadores de *hosts* são divididos em dois escopos desenvolvidos por uma DHT local, responsável pelos mapeamentos dentro de um sistema autônomo; e uma DHT global, responsável pelo mapeamento no núcleo da topologia da rede. Do mesmo modo, tal abordagem propõe a adição de novas camadas conforme a necessidade hierárquica do sistema.

Além disso, os protocolos DHTs formam topologias sobrepostas independentes das redes físicas existentes. Isto implica que os mapeamentos distribuídos em uma topologia DHT não dependem da topologia de rede física, podendo ser distribuídos em vários sistemas autônomos, bem como organizados em hierarquias de acordo com a necessidade do escopo a ser alcançado.

4.7 Localização e recuperação de conteúdo

Nesta seção apresenta-se um estudo de caso genérico utilizando a arquitetura proposta para delinear o processo de localização e recuperação de um objeto de conteúdo no contexto das redes centradas na informação. Neste contexto, todo o relacionamento entre os *hosts* e os mecanismos de resolução é baseado no paradigma publica/assina, no qual os *hosts* publicam a posse das entidades nos mecanismos de resolução, e requisitam valores associados às chaves através de assinaturas de entidades.

A **Figura 30** ilustra publicações e assinaturas das entidades envolvidas no processo de localização e recuperação de um objeto de conteúdo.

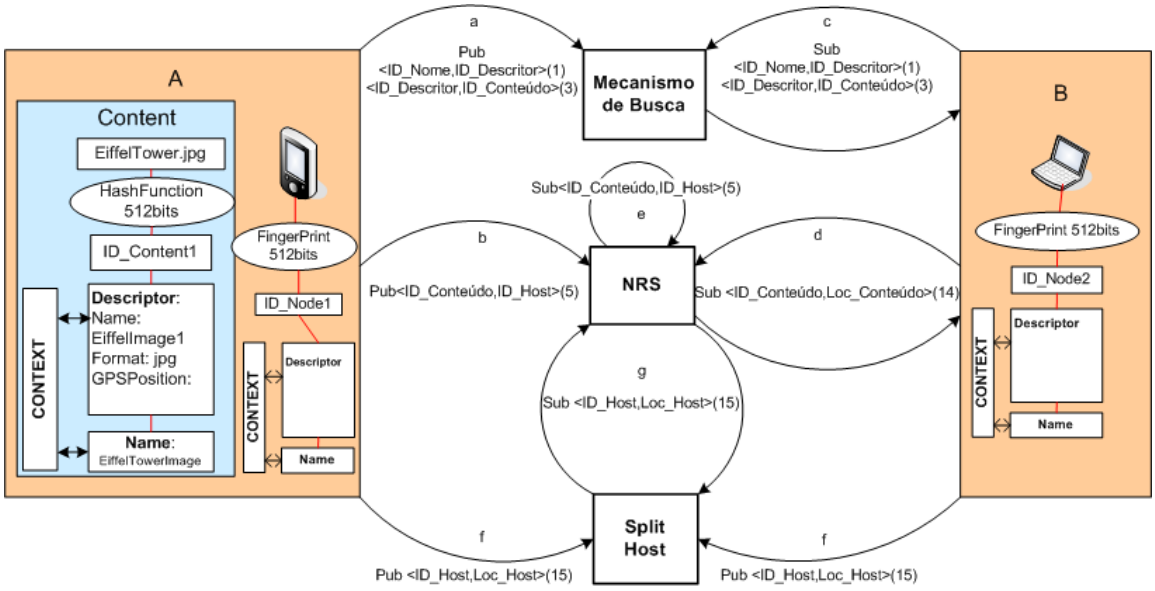

Figura 30. *Relacionamento publica/assina do GIRS para a localização e recuperação de conteúdo.*

Considere o cenário ilustrado pela Figura 30, no qual existe um nó (B) interessado em um determinado conteúdo (*EiffelTower.jpg*), que está disponibilizado remotamente em outro nó (A). Neste ambiente, por padrão, todo conteúdo a ser publicado passa por uma série de interações responsáveis por identificar este conteúdo, além de descrevê-lo e nomeá-lo. Para diminuir a complexidade deste exemplo, estas interações não estão presentes nesta figura.

Após estas interações, o nó responsável pela publicação (ou até mesmo um nó *cache*) do conteúdo deve publicar o nome e o descritor junto ao mecanismo de busca (passo *a*, através dos mapeamentos 1 e 3 listados na Tabela 4), bem como informar a posse do conteúdo ao sistema de resolução de nomes (NRS) (passo *b,* através do mapeamento 5 da Tabela 4).

Dado que o conteúdo interessado já está publicado no mecanismo de busca e no NRS, o nó assinante procura pelo identificador único do conteúdo no sistema de busca baseando-se no nome deste conteúdo e/ou nos atributos e metadados contidos no seu descritor (passo *c*, utilizando também os mapeamentos 1 e 3 da Tabela 4). Se houver resultados disponíveis, estes são retornados para o usuário do nó (B) na forma de descritores e/ou nomes legíveis. Então, com base nestas informações legíveis, o usuário faz um refinamento da busca e elege um resultado com um identificador para o conteúdo interessado. Este identificador é

inserido como um parâmetro para a assinatura do conteúdo no NRS (passo *d,* relacionado ao mapeamento 14 da Tabela 4).

No NRS é feito o mapeamento entre o identificador do conteúdo e o identificador do nó responsável por esse conteúdo (passo *e,* relacionado ao mapeamento 5 da Tabela 4). Desta forma, o NRS pode transmitir diretamente a mensagem de *Sub<Conteúdo,ID_Conteúdo>* para o nó publicador através do encaminhamento direto ou devolver o resultado da resolução (ou seja, os localizadores dos objetos resolvidos) ao assinante, tal como acontece no NetInf (OHLMAN, et al., 2009).

Entretanto, antes de devolver o localizador do nó publicador ao nó assinante, o NRS interage com algum mecanismo de desacoplamento de identificação e localização de *hosts,* a fim de fazer o mapeamento dinâmico entre o identificador e o localizador do nó publicador (passo *f* e *g,* desenvolvidos através do mapeamento 15 da Tabela 4), e desta forma, suportar a mobilidade e *multihoming* dos *hosts*. É importante ressaltar que os passos *f* e *g* são considerados assíncronos, uma vez que tais requisições podem ser efetuadas a qualquer momento, ou sempre que for solicitado o localizador atualizado de um *host.*

Note, portanto, que todo conteúdo a ser disponibilizado é identificado através de uma função *hash* deste conteúdo. O identificador formado pela função *hash* é opaco e livre de semântica, ou seja, não é legível, nem possui significado próprio. Entretanto, nesta proposta, cada conteúdo possui um descritor contendo atributos responsáveis por descrevê-lo e auxiliar aos mecanismos de busca na sua localização. Da mesma forma, cada conteúdo possui um nome legível baseado no contexto deste objeto descrito em linguagem natural, que também é publicado nos mecanismos de busca.

Dado que os nomes planos são melhores para implementações de segurança e nomes hierárquicos geralmente são criados para serem legíveis, esta proposta possui as vantagens das duas arquiteturas de nomeação. Em outras palavras, o GIRS separa a identificação da nomeação das entidades, sendo que os identificadores são criados através de uma função *hash*, sendo planos e opacos, e os nomes e descritores são legíveis com base na semântica e no contexto do conteúdo.

Esta abordagem propõe também que os *hosts* sejam identificados de forma semelhante à utilizada para identificação de conteúdo. Os identificadores de *hosts* podem ser criados com base nas informações únicas de cada dispositivo, criando uma espécie de impressão digital do *hardware*, permitindo relações entre entidades de diversas espécies, como conteúdos e dispositivos. Da mesma forma, estes *hosts* podem ter nomes legíveis e descritores ligados a seus identificadores.

4.8 Especificação de *Software*

A UML (*Unified Modeling Language*) é uma linguagem que descreve os processos do desenvolvimento de sistemas através de um conjunto de diagramas. Tais diagramas podem ser utilizados para modelar *softwares* no ponto de vista estrutural (estático) e comportamental (dinâmico) (PENDER, 2004). Nas seções seguintes serão apresentados alguns diagramas UML para a especificação de *software* do GIRS.

4.8.1 Diagramas de Casos de Uso

Os diagramas de Casos de Uso (*Use Case*) são diagramas UML comportamentais responsáveis por modelar as expectativas dos usuários que irão utilizar o sistema. A **Figura 31** ilustra o diagrama de Casos de Uso do GIRS para a publicação e assinatura de conteúdos.

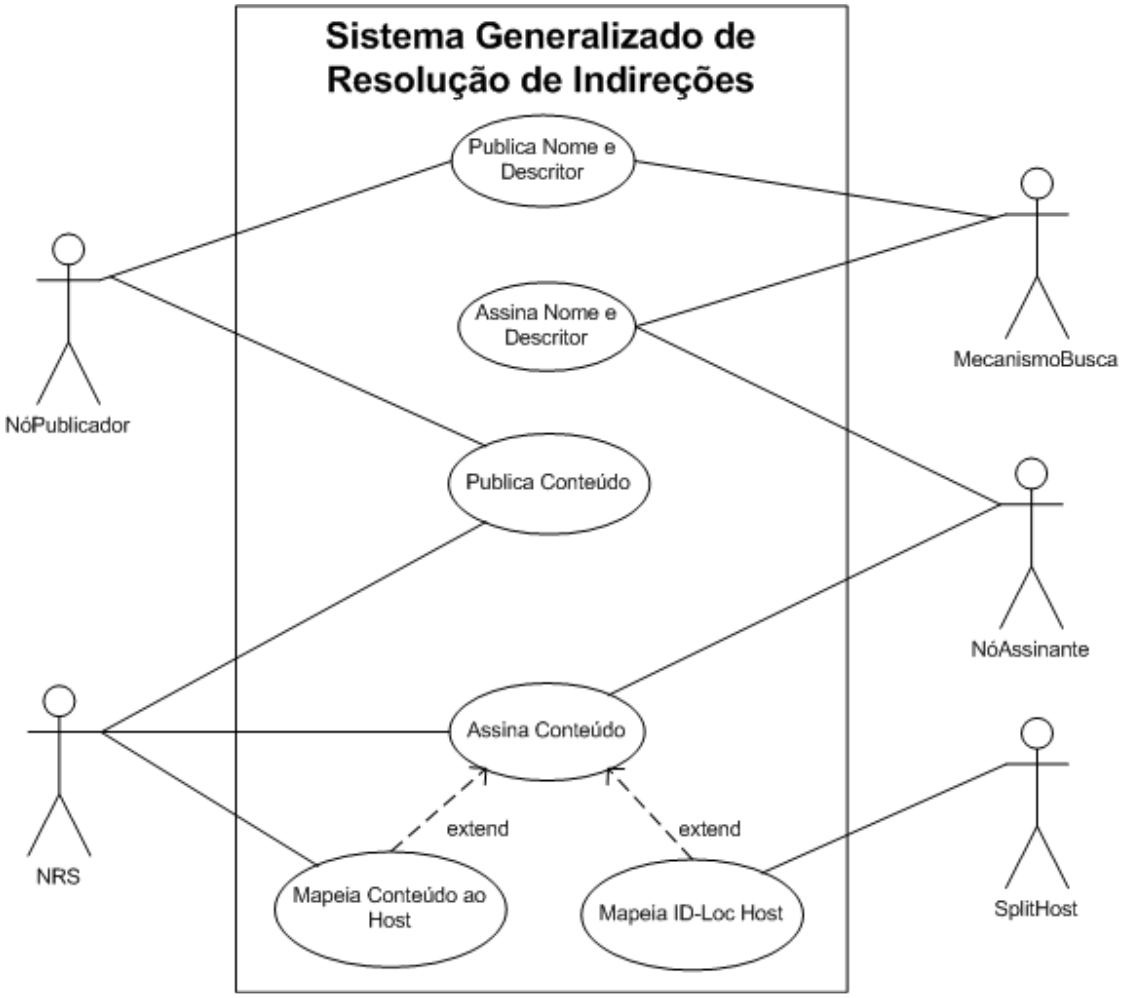

Figura 31. *Diagrama de casos de uso do GIRS para a publicação e assinatura de conteúdos.*

A Figura 31 ilustra a interação entre os atores *NóPublicador* e *NóAssinante* que representam os usuários do sistema com os mecanismos de resolução *NRS* (*Name Resolution System*); o *Mecanismo de Busca SE* (*Search Engine*) e o *SplitHost* (mecanismo de desacoplamento de identificadores e localizadores de *hosts*). Estes mecanismos de resolução são também considerados atores, uma vez que eles interagem externamente com o sistema. Ainda na Figura 31, cada caso de uso representa uma interação entre os atores, onde o caso de uso:

- *Publica Nome e Descritor:* representa a publicação do nome e do descritor relacionado a um conteúdo feito pelo *NóPublicador* junto ao *Mecanismo de Busca.* Estas publicações são desenvolvidas utilizando os mapeamentos do tipo 1 e 3, correspondentes ao passo *a* da Figura 30;

- *Publica Conteúdo:* representa a publicação do identificador de um conteúdo feito pelo *Nópublicador* junto ao *Sistema de Resolução de Nomes.* Esta publicação é feita utilizando o mapeamentos do tipo 5, correspondente ao passo *b* da Figura 30;

- *Assina Nome e Descritor*: representa a assinatura do nome e/ou descritor de um conteúdo feito pelo *NóAssinante* junto ao *Mecanismo de Busca* (passo *c* da Figura 30, utilizando os mapeamentos do tipo 1 e 3);

- *Assina Conteúdo:* representa a assinatura de um conteúdo através de seu identificador feito pelo *NóAssinante* junto ao *Sistema de Resolução de Nomes* (passo *d* da Figura 30, utilizando o mapeamentos do tipo 14);

- *Mapeia Conteúdo ao Host:* representa uma interação estendida do caso de uso *Assina Conteúdo* desenvolvida pelo *Sistema de Resolução de Nomes.* Este caso de uso é responsável por mapear o identificador do conteúdo ao identificador do *host* responsável pelo armazenamento deste conteúdo (passo *e* da Figura 30, utilizando o mapeamento do tipo 5). A interação deste caso de uso com o caso de uso *Assina Conteúdo* é feita através de um relacionamento <<*extend*>>, uma vez que o caso de uso *Mapeia Conteúdo ao Host* pode ser acionado quando o *NóAssinante* não possuir o identificador do *host* responsável por armazenar o conteúdo requisitado;

- *Mapeia ID-Loc Host:* representa também uma interação estendida do caso de uso *Assina Conteúdo,* sendo desenvolvida, entretanto, pelo *SplitHost.* Este caso de uso é responsável por mapear o identificador no localizador do *host* responsável pelo armazenamento do conteúdo (passo *f* e *g* da Figura 30, utilizando o mapeamento do tipo 15). A interação deste caso de uso também interage com o caso de uso *Assina Conteúdo*, através do relacionamento <<*extend*>> e não precisa ser acionado uma vez que o nó assinante já possui o localizador do nó publicador.

4.8.2 Diagramas de Seqüência

Os diagramas de Seqüência formam outra espécie de diagramas UML comportamentais e são responsáveis por modelar as interações entre os objetos em função do tempo em que estas interações ocorrem.

A **Figura 32** ilustra a interação dos nós (assinante e publicador de um conteúdo) com os mecanismos de resolução (Mecanismo de Busca, Sistema de Resolução de Nomes e Separador de Identificador e Localizador de *Hosts*) através de um diagrama de Seqüência.

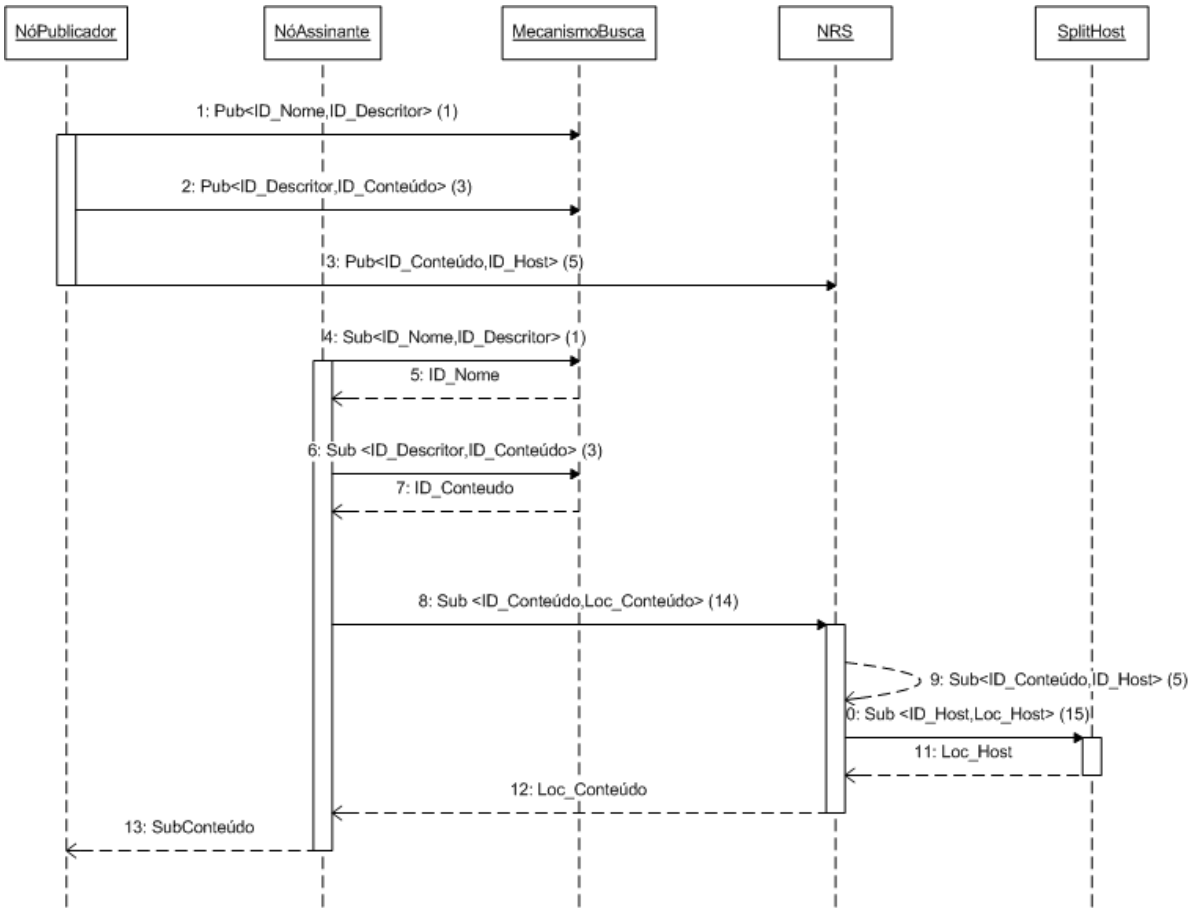

Figura 32. *Diagrama de seqüência do GIRS para a publicação e assinatura de conteúdos.*

Para a Figura 32, as mensagens 1, 2 e 3 correspondem à publicação de um conteúdo, enquanto as mensagens de 4 a 13 correspondem à assinatura de um conteúdo publicado. Sendo que a mensagem:

1. *Pub<ID_Nome,ID_Descritor>:* representa a publicação do descritor relacionado a um nome de conteúdo, feita pelo *NóPublicador* junto ao *Mecanismo de Busca* (passo *a* da Figura 30, desenvolvido através de um mapeamento do tipo 1);
2. *Pub<ID_Descritor,ID_Conteúdo>:* representa a publicação do identificador de um conteúdo relacionado ao identificador do seu descritor. Esta publicação é feita pelo *NóPublicador* junto ao *Mecanismo de Busca* (passo *a* da Figura 30, desenvolvido através de um mapeamento do tipo 3);
3. *Pub<ID_Conteúdo,ID_Host>:* representa a publicação do identificador de um *host* responsável por armazenar o conteúdo relacionado. Esta publicação é desenvolvida pelo *NóPublicador* junto ao *Sistema de Resolução de Nomes* (passo *b* da Figura 30, desenvolvido através de um mapeamento do tipo 5);
4. *Sub<ID_Nome,ID_Descritor>:* representa a assinatura do descritor relacionado a um nome de conteúdo feita pelo *NóAssinante* junto ao *Mecanismo de Busca* (passo *c* da Figura 30, desenvolvido através de um mapeamento do tipo 1);
5. *ID_Descritor:* corresponde à mensagem retornada pela requisição de assinatura do descritor desenvolvida pela mensagem 4;
6. *Sub <ID_Descritor,ID_Conteúdo>:* representa a assinatura do identificador de um conteúdo relacionado a seu descritor feita pelo *NóAssinante* junto ao *Mecanismo de Busca* (passo *c* da Figura 30, desenvolvido através de um mapeamento do tipo 3);
7. *ID_Conteúdo:* corresponde à mensagem retornada pela requisição de assinatura do identificador do conteúdo desenvolvida pela mensagem 6;
8. *Sub <ID_Conteúdo,Loc_Conteúdo>:* representa a assinatura do localizador de um conteúdo a partir do seu identificador feita pelo *NóAssinante* junto ao *Sistema de Resolução de Nomes* (passo *d* da Figura 30, desenvolvido através de um mapeamento do tipo 14);
9. *Sub<ID_Conteúdo,ID_Host>:* corresponde a uma mensagem de auto-referência através da qual o *Sistema de Resolução de Nomes* faz internamente o mapeamento entre o identificador do conteúdo requisitado e o identificador do *host* responsável pelo armazenamento deste conteúdo (passo *e* da Figura 30, desenvolvida através de um mapeamento do tipo 5);

10. *Sub <ID_Host,Loc_Host>:* corresponde a uma assinatura do localizador do *host* a partir de seu identificador feita pelo *Sistema de Resolução de Nomes* junto ao *SplitHost* (passo *g* da Figura 30 desenvolvida através de um mapeamento do tipo 15);
11. *Loc_Host:* corresponde a uma mensagem com o localizador do *host* solicitado pela mensagem 10;
12. *Loc_Conteúdo:* corresponde a uma mensagem retornada ao *NóAssinante*, contendo o localizador do conteúdo requisitado;
13. *SubConteúdo:* corresponde à assinatura do conteúdo, propriamente dito, solicitada pelo *NóAssinante* ao *NóPublicador*.

4.8.3 Diagramas de Atividades

Os diagramas de Atividades são diagramas comportamentais UML que modelam a lógica exigida para implementar os comportamentos do sistema. Este tipo de diagrama ilustra o fluxo de trabalho através de tarefas e decisões a serem executadas pelo usuário e/ou sistema (PENDER, 2004). A **Figura 33** ilustra o diagrama de Atividades do GIRS para a assinatura de conteúdos.

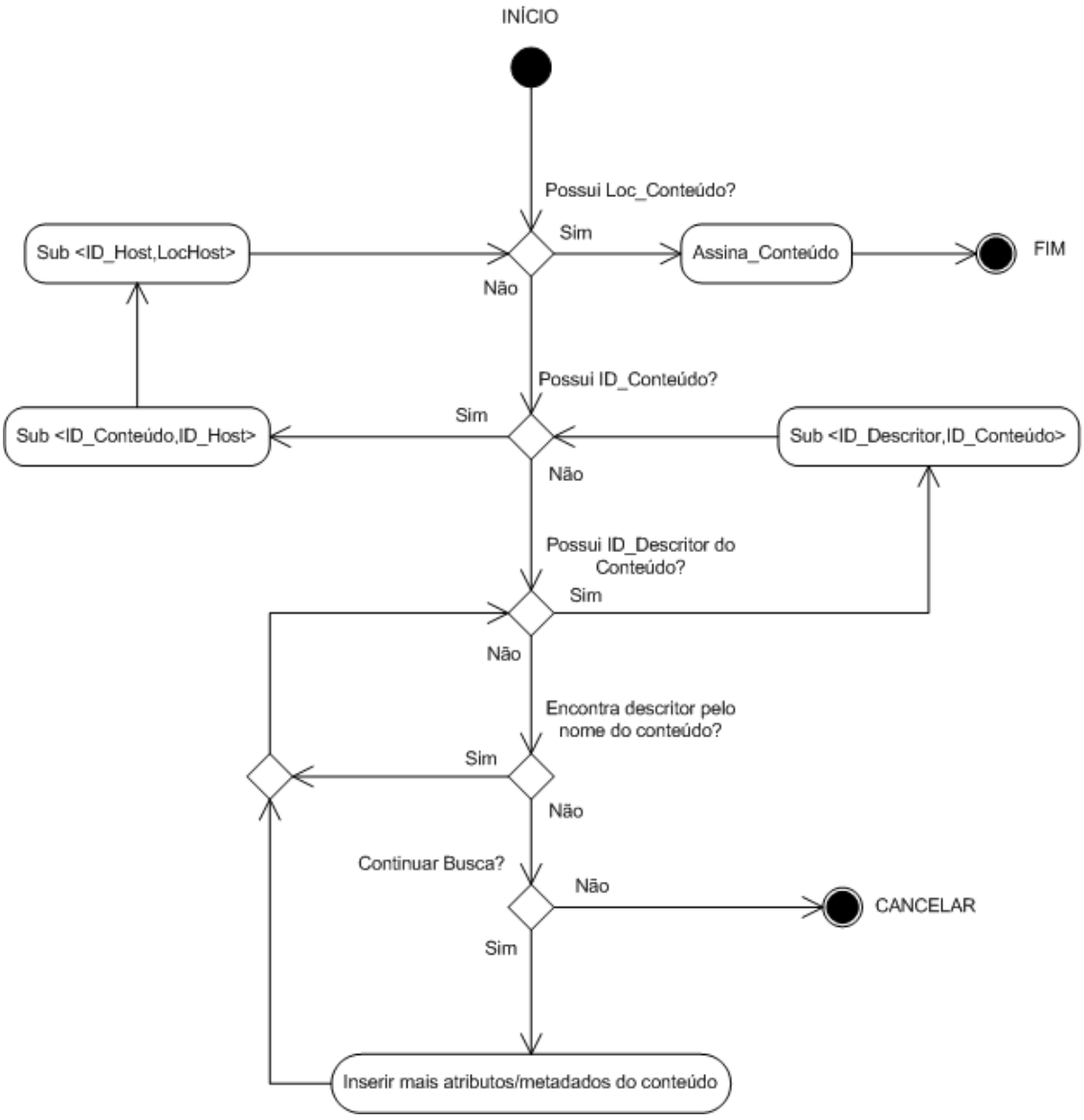

Figura 33. *Diagrama de atividades do GIRS para a publicação e assinatura de conteúdos.*

A Figura 33 ilustra o fluxo de decisões e de tarefas para a assinatura de um conteúdo do ponto de vista do usuário. Para este diagrama, considera-se que o conteúdo já se encontra publicado no *Mecanismo de Busca*, bem como no *Sistema de Resolução de Nomes.*

Uma vez que o usuário já possui o localizador do conteúdo (ou o localizador do *host* responsável por armazenar o conteúdo) de interesse, o fluxo é então direcionado para a atividade *Assina_Conteúdo* e, posteriormente, o fluxo é então finalizado. Caso contrário, o fluxo é direcionado para mapeamentos seqüenciais até que o usuário consiga tal localizador.

Caso o usuário não possua o identificador do conteúdo de interesse, o fluxo é então direcionado para pesquisas baseadas em informações legíveis que auxiliam o processo de busca do descritor e, posteriormente, do identificador deste conteúdo. Existe ainda a possibilidade de o usuário cancelar a busca, bem como reiniciá-la, inserindo ou refinando os atributos e metadados organizados em descritores do conteúdo.

4.8.4 Diagrama de Classes

Conforme (PENDER, 2004), os diagramas de Classes são diagramas estruturais UML responsáveis por modelar os recursos e os relacionamentos entre estes recursos utilizados pelo sistema. Recursos podem representar qualquer entidade que possua atributos e operações como: pessoas, materiais, *hosts* e conteúdos. Estes diagramas são considerados um dos principais diagramas UML, uma vez que eles modelam os recursos essenciais para a operação correta do sistema, além de serem os principais diagramas utilizados para a geração do código do sistema.

A **Figura 34** ilustra o diagrama de Classes sob uma visão ampla das classes do GIRS para a publicação e assinatura de conteúdos.

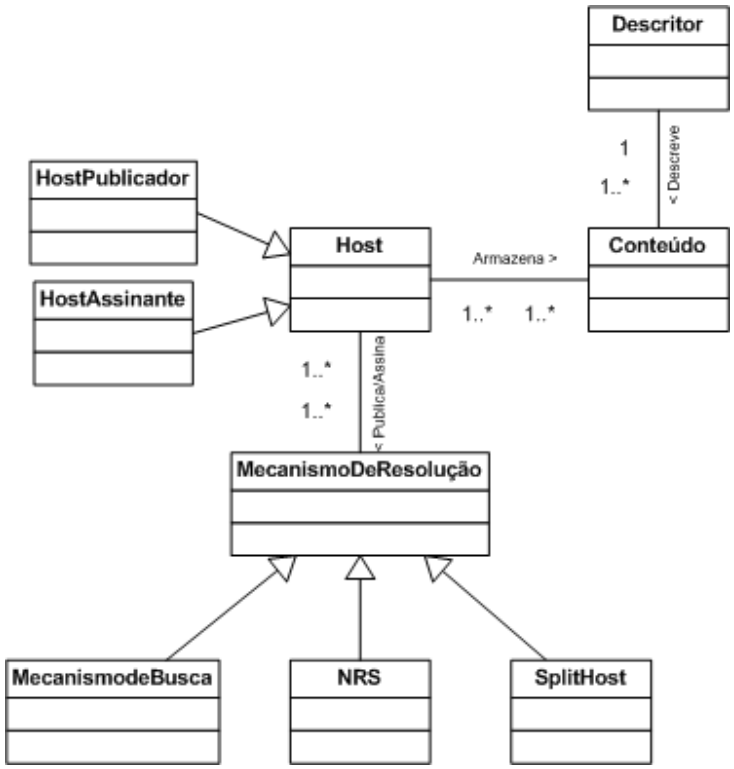

Figura 34. *Diagrama de classes (a) do GIRS para a publicação e assinatura de conteúdos.*

A classe *Host* é uma generalização das classes *HostPublicador* e *HostAssinante*, dado que ambas possuem atributos em comum. Além de armazenar objetos da classe *Conteúdo*, os objetos da classe *Host* publicam e assinam objetos de *Conteúdo* junto aos objetos da classe *MecanismodeResolução*. Já a classe *MecanismodeResolução* generaliza as classes *MecanismodeBusca, NRS* e *SplitHost*, uma vez que ambas compartilham a mesma finalidade, que é aceitar assinaturas e publicações. Por outro lado, os objetos da classe *Descritor* são responsáveis por descrever objetos da classe *Conteúdo*.

A **Figura 35** ilustra de forma detalhada os atributos, operações e relacionamentos das classes do GIRS para a publicação e assinatura de conteúdos.

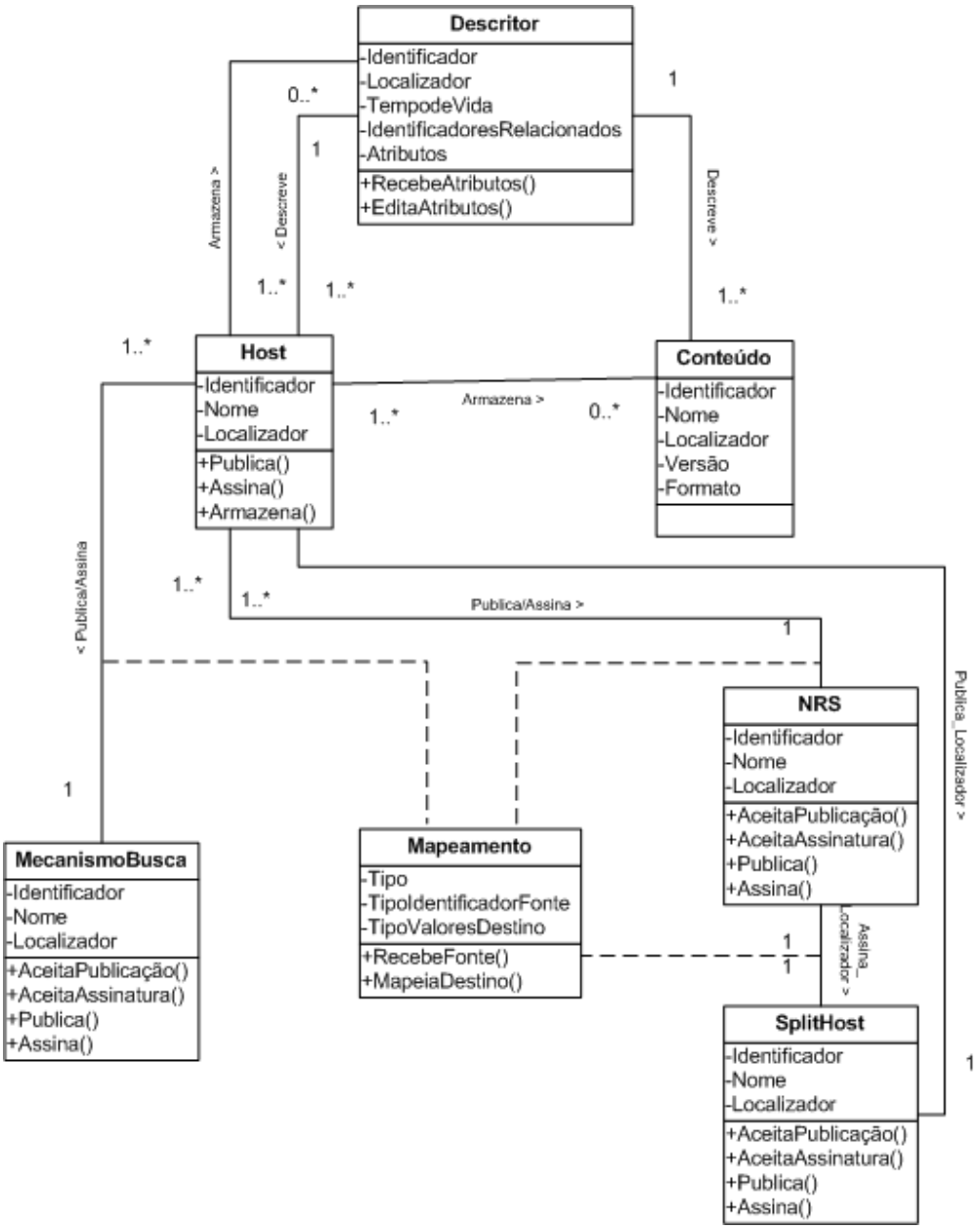

Figura 35. *Diagrama de classes (b) do GIRS para a publicação e assinatura de conteúdos.*

Os objetos da classe *Host* possuem os atributos *Identificador, Nome e Localizador,* bem como a operação *Armazena,* através da qual eles se relacionam com os objetos das classes *Conteúdo e Descritor*. Já as operações *Publica e Assina* da classe *host* representam os relacionamentos entre esta classe e os *MecanismosdeResolução* (*Mecanismo de Busca, NRS* e o *SplitHost*).

Conforme citado na Seção 4.2 e ilustrado pela Figura 34, apesar de existir a especialização das classes *MecanismodeBusca, NRS* e o *SplitHost*, ambas poderiam ser generalizadas e uma mesma classe *MecanismodeResolução* teria os mesmos atributos e desempenharia as operações das três classes especializadas. Entretanto, justifica-se a especialização de tais classes, o fato de elas usarem tipos de mapeamentos peculiares a cada classe e, para reduzir a complexidade do projeto, dividindo as classes de resolução em módulos.

Os objetos da classe *Descritor* descrevem os objetos da classe *Conteúdo* através das operações *RecebeAtributos e EditaAtributos.* A operação *RelacionaIdentificador*, como o próprio nome diz, é responsável por relacionar identificadores de objetos da classe *Conteúdo* descritos por um objeto da classe *Descritor.* Assim como mostrado na estrutura do descritor, apresentado na Seção 4.3, os objetos da classe *Descritor* possuem os atributos *Identificador, Localizador, TempodeVida* e *IdentificadoresRelacionados.*

Os objetos da classe *Conteúdo* possuem os atributos *Identificador, Nome* e *Atributos*, sendo que este último representa informações sobre o objeto de conteúdo como versão, codificação e formato de arquivo. Já a classe *Mapeamentos* corresponde a uma classe de associação, uma vez que os objetos da classe *Host* utilizam estas classes para publicar e assinar os atributos dos objetos de conteúdo junto aos *Mecanismos de Resolução.*

Capítulo 5

Conclusões

Nesta dissertação foram reunidas diversas arquiteturas de redes que propõem a separação entre identificadores e localizadores de *hosts* e da informação. A partir deste levantamento, foram feitas algumas comparações com o objetivo de delinear os principais desafios para se aplicar tal desacoplamento. Além disso, verificaram-se as principais vantagens e desvantagens dos sistemas planos e hierárquicos de identificação utilizados por arquiteturas de redes atuais e futuras.

Concluiu-se que, para identificar independentemente e unicamente as entidades (*hosts* e objetos de conteúdo), os sistemas de identificadores planos são extremamente eficientes, uma vez que tais sistemas não são atrelados a entidades centralizadas para criação e gerenciamento destes identificadores. Este fato torna os identificadores livres de qualquer estrutura hierárquica baseada em localização, suportando naturalmente o desacoplamento de identificadores e localizadores destas entidades e mais seguros, uma vez que os sistemas de identificação não possuem um ponto centralizado de vulnerabilidade.

Entretanto, identificadores planos geralmente são opacos e livres de semântica. Desta forma, além da necessidade de identificar unicamente entidades (*hosts* e objetos de conteúdo), é necessário nomear e descrever tais entidades de forma legível. Com isso, propõe-se a atribuição de nomes e de objetos descritores legíveis aos *hosts* e aos objetos de conteúdo, de forma que as informações legíveis possam auxiliar o processo de busca e localização das entidades com base na sua semântica e no seu contexto.

Para o mapeamento de identificadores planos, as DHTs formam uma estrutura descentralizada e altamente escalável. Isto se deve à natureza distribuída e genérica dos protocolos DHTs, o que permite criar redes sobrepostas independentes da arquitetura de rede existente, formando topologias eficientes para distribuir servidores de mapeamentos ao longo da rede.

Com base nas análises, comparações e nos desafios delineados entre as arquiteturas de redes estudadas, foi proposto um Sistema Generalizado de Resolução de Indireções para tratar os mapeamentos dinâmicos entre os atributos (nome, identificador, localizador e descritor) de *hosts* e de objetos de conteúdo. Justifica-se tal proposta pela necessidade de identificar unicamente as entidades, bem como nomeá-las e descrevê-las de forma legível, além de criar ligações dinâmicas entre os seus atributos, suportando naturalmente a mobilidade e o *multihoming* destas entidades.

Em comparação com cada uma das arquiteturas estudadas, as principais vantagens ao se utilizar o Sistema Generalizado de Resolução de Indireções são:

- desacoplamento de identificadores e localizadores das entidades físicas e lógicas, suportando naturalmente a mobilidade e o *multihoming* destas entidades;
- identificação estatisticamente única baseada em identificadores planos, em conjunto com nomes e descritores criados com informações legíveis e semântica para *hosts* e conteúdos;
- resolução de indireções para identificadores, nomes, descritores e localizadores de forma generalizada através de mapeamentos dinâmicos e distribuídos;
- independência de tecnologia de interconexão. Pode-se aplicar às redes IP atuais, utilizando um dos protocolos DHTs, bem como para abordagens de redes centradas na informação;

- flexibilidade da arquitetura para a inserção de novas entidades físicas como pessoas e usuários e de entidades lógicas como serviços, domínios e aplicações, bem como a flexibilidade para a inserção de novas classes de mapeamentos relacionados às entidades inseridas.

Em suma, as principais contribuições deste trabalho estão na análise do problema para a identificação de entidades em redes atuais e futuras e uma proposta para a solução dos problemas delineados. Visto que o problema da identificação e da localização de *hosts* e de objetos de informação é muito amplo, algumas sugestões de trabalhos futuros incluem:

- desenvolvimento de um código de implementação de *software* para o Sistema Generalizado de Resolução de Indireções, incluindo o código para a estrutura dos descritores baseado em uma linguagem que forneça informações legíveis, como por exemplo, a XML ou a OWL;

- desenvolvimento de um ambiente de testes para o Sistema Generalizado de Resolução de Indireções utilizando uma das abordagens DHTs que implemente o cenário de mapeamentos com o paradigma publica/assina;

- ampliação da análise e agregação à proposta de técnicas de segurança, como por exemplo a auto-certificação da integridade dos dados e autenticação do proprietário propostas em (AHLGREN, et al., 2010); mecanismos de escopo e privacidade propostos em (AIN, et al., 2010), múltiplos domínios e níveis hierárquicos propostos em (HANKA, et al., 2008);

- ampliação do estudo e análise de técnicas de roteamento baseado em identificadores planos como o ROFL (CAESAR, et al., 2006), o LLC (OHLMAN, et al., 2009), o SPSwitch (ESTEVE, et al., 2008) e o roteamento plano baseado na operação XOR (PASQUINI, et al., 2010b).

Referências Bibliográficas

HARAI, hiroaki. et al. **Akari Project (2007).** Web site. Website.://akari-project.nict.go.jp/eng/concept-Design/AKARI_fulltext_e_translated_version_1_1.pdf. 2007.

STANFORD UNIVERSITY. **Clean Slate: An Interdisciplinary Research Program.** Disponível em: http://cleanslate.stanford.edu/about_cleanslate.php. Último Acesso: 24/08/2010.

BELLOVIN Steven M., CLARK David D., SONG Adrian Perrig Dawn. A Clean-Slate Design for the Next-Generation Secure Internet. Relatório técnico, Pittsburgh, PA: Report for NSF Global Environment for Network Innovations (GENI) Workshop.. Julho de 2005.

IN Min-kyo, LEE Seung-yun, KIM Dae-young. **Splitting mechanism for IP into Identifier and Locator in NGN.** The 9th International Conference on Advanced Communication Technology, 2007.

JACOBSON Van, SMETTERS Diana K., THORNTON James D., PLASS Michael F. BRIGGS Nicholas H., BRAYNARD Rebecca L. **Networking Named Content.** Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies 2009.

PALO ALTO RESEARCH CENTER INCORPORATED. **Focus Area.** Disponível em: http://www.parc.com/work/focus-area/networking/. Último Acesso: 24/06/2010.

ESTEVE Christian, VERDI L. Fábio, MAGALHÃES F. Maurício. **Towards a New Generation of Information-Oriented Internetworking Architectures.** Proceedings of the 2008 CoNEXT Conference. 2008.

BIHAM, Eli; DUNKELMAN, Orr. **\The SHAvite-3 Hash Function, Submissionto NIST (Round 2)".** Available: http://csrc.nist.gov/groups/ST/hash/sha-3/Round2/documents/SHAvite-3_Round2.zip. 2009.

STEVENS, Marc. **On Collisions for MD5**. Dissertação de Mestrado: Eindhoven University of Technology, The Netherlands. http://www.win.tue.nl/hashclash/On%20Collisions%20for%20MD5%20-%20M.M.J.%20Stevens.pdf). 2007.

KOZIEROK, Charles **The TCP/IP Guide: A Comprehensive, Illustrated Internet Protocols Reference –** Book. Starch Press. San Francisco: 2005.

MOCKAPETRIS, P. **RFC 1034 - Domain Names - Concepts and Facilities**. 1987.

AHLGREN Bengt, D'AMBROSIO Matteo, DANNEWITZ Christian. **Second NetInf Architecture Description.** FP7 4WARD Project Deliverable 6.2. Disponível em: www.4ward-project.eu/index.php?id=192. 2010.

PERKINS C. **RFC3344 - IP Mobility Support for IPv4.** 2002.

RAMACHANDRAN, Kishore. **Mobile IP - deployment after a decade.** 2005.

BALDO, Jardel Pavan. **Mobile IP X Hip: Um Estudo Sobre Segurança Em Redes Móveis. Monografia**. 2007.

JOHNSON D., PERKINS C. and ARKKO J., **Mobility Support in IPv6**, RFC 3775. 2004.

BARBATO, Wander. **A mobilidade na internet com o padrão HIP**. Vol. 2, No 2 (2007): Revista de Ciências Exatas e Tecnologia 2007.

MOSKOWITZ, R. NIKANDER, P. **Host Identity Protocol (HIP) Architecture RFC 4423.** Maio 2006.

JIANLI Pan, SUBHARTHI Paul, RAJ Jain, MIC Bowman. **MILSA: A Mobility and Multihoming Supporting Identifier Locator Split Architecture for Naming in the Next Generation Internet.** IEEE GLOBECOM 2008, New Orleans, LA, December 2008.

LEWIS, D.; MEYER, D.; FARINACCI, D.; FULLER, V. **Locator/ID Separation Protocol (LISP).** Draft. Work in Progress. 2010. Disponível em http://tools.ietf.org/html/draft-ietf-lisp-06 Último acesso em 20/04/2010 LISP Draft 06, 25 de janeiro de 2010.

IANNONE, L., SAUCEZ, D., BONAVENTURE, O., **OpenLISP: An Open Source Implementation of the Locator/ID Separation Protocol.** IEEE Infocom 2009.

MEYER, D. **The Locator/Identifier Separation Protocol (LISP).** The Internet Protocol Journal, Volume 11, No. 1. Disponível em http://www.cisco.com/web/about/ac123/ac147/archived_issues/ ipj_11-1/111_lisp. html. Último acesso em 20 de abril de 2010.

CAESAR Matthew, CONDIE Tyson, KANNAN Jayanthkumar, LAKSHMINARAYANAN Karthik, STOICA Íon and SHENKER Scott. **ROFL: Routing on Flat Labels.** Proceedings of the 2006 conference on Applications, Technologies, Architectures, and Protocols for Computer Communications 2006.

CAMPISTA, Miguel Elias, et al., **Interconexão de Redes na Internet do Futuro: Desafios e Soluções.** 2010. Disponível em: http://www.gta.ufrj.br/ftp/gta/TechReports/CFM10.pdf. Último acesso em 15 set. 2010.

NIEBERT Norbert, LUNDGREN Lars and ABRAMOWICZ Henrik . FP7 4WARD Project Deliverable 0.1, **Dissemination and Expoitation Plan.** Disponível em: www.4ward-project.eu/index.php?s=file_download&id=45. 2008.

DANNEWITZ, Christian. **NetInf: An Information-Centric Design for the Future Internet.** Proc. 3rd GI/ITG KuVS Workshop, May 2009.

LIU Shi, BI Jun, WANG Yangyang. **A DHTs-Based Mapping System for Identifier and Locator.** First International Conference on Advances in Future Internet. 2009

DANNEWITZ, Christian, GOLIC Jovan, OHLMAN Borje, AHGREN Bengt. **Secure Naming for a NetInf**. INFOCOM IEEE Conference on Computer Communications Workshops. 2010.

AHLGREN Bengt, D'AMBROSIO Matteo, MARSH Ian, MARCHISIO Marco, STRANDBERG Ove. **Design Considerations for a Network of Information.** Proceedings of the 2008 ACM CoNEXT Conference. 2008.

OHLMAN Börje, et al., **First NetInf Architecture Description.** FP7 4WARD Project Deliverable 6.1. Disponível em: www.4ward-project.eu/index.php?s=file_ download&id=39. 2009.

WONG Walter, VERDI Fábio and MAGALHÃES Maurício F. **A Security Plane for Pub-Sub Based Content Oriented Networks.** CoNEXT '08 Proceedings of the 2008 ACM CoNEXT Conference. 2008.

ZAHEMSZKY András, GAJIC Borislava, ESTEVE Christian Rothenberg. **Experimentally driven research in Publish/Subscribe Information-Centric InterNetworking.** In Tridentcom, Berlin, Germany. 2009.

AIN Mark, et al., **PSIRP Publish-Subscribe Internet Routing Paradigm Deliverable 2.2: Conceptual Architecture of PSIRP Including Subcomponent Descriptions.** Disponível em: psirp.hiit.fi/.../Deliverables/FP7-INFSO-ICT-216173-PSIRP-D2.2_ ConceptualArchitecture_v1.1.pdf. 2008.

PASQUINI Rafael, MAGALHÃES Maurício F., VERDI Fábio L. and WELIN Annikki. **Bloom filters in a Landmark-based Flat Routing.** In 2010 International Conference on Communications. 2010-a.

NIKANDER Pekka and MARIAS Giannis F., Ericsson Research. **Towards Understanding Pure Publish/Subscribe. Cryptographic Protocols.** Cambridge Security Protocols Workshop (SPW 2008). 2009.

ESTEVE Christian. **Towards Content-Centric Internetworking Based on The Publisher Subscriber Paradigm.** Apresentação. 2008.

BRODER Andrei, MITZENMACHER Michael. **Network Applications of bloom filter.** In Internet Mathematics Vol.1 No. 4. 2004.

FULLER, V. LI, T. **Classless Inter-domain Routing (CIDR): The Internet Address Assignment and Aggregation Plan.** RFC-4632. 2006.

MINTS - Minnesota Internet Traffic Studies (MINTS). **The exabyte era.** Acessado em: http://www.dtc.umn.edu/mints/references.html. Último acesso em: 20 de setembro de 2010.

STOICA, Ion, et al. **Internet Indirection Infrastructure.** In IEEE/ACM Transactions on Networking (TON). 2002.

WIKIPEDIA. **Indirection**, 2010. Disponível em: http://en.wikipedia.org/ wiki/Indirection. Último acesso em: 20 de setembro de 2010.

WIKIPEDIA. **Distributed Hash Table**. Disponível em http://pt.wikipedia.org/wiki/ Distributed_hash_table. Último acesso em: 14 de setembro de 2010.

STOICA Ion, MORRIS Robert, KARGER David, KAASHOEK M. Frans, BALAKRISHNAN Hari. **Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications.** In IEEE/ACM Transactions on Networking (TON). 2003.

MAYMOUNKOV Petar and MAZIERES David. **Kademlia: A Peer-to-peer Information System Based on the XOR Metric.** Workshop on Peer-to-Peer Systems. 2002.

CIRANI Simone Cirani and VELTRI Luca Veltri. **Implementation of a framework for a DHT-based Distributed Location Service.** In 16th International Conference on Software, Telecommunications and Computer Networks. 2007.

GHODSI Ali. **Distributed k-ary System: Algorithms for Distributed Hash Tables.** Ph.D Thesis. 2006.

LUA Eng Keong, CROWCROFT Jon, PIAS Marcelo, SHARMA Ravi and LIM Steven. **A Survey and Comparison of Peer-to-Peer Overlay Network Schemes**. In IEEE Journals. 2005.

HANKA Oliver, SPLEIB Christoph, KUNZMANN Gerald, EBERSPACHER Jorg. **A DHT-inspired clean-slate approach for the Next Generation Internet.** In 2nd GI/ITG KuVS Workshop on The Future Internet. 2008.

RIECHE Simon, WEHRLE Klaus, LANDSIEDEL Olaf, GOTZ Stefan, PETRAK Leo. **Reliability of Data in Structured Peer-to-Peer Systems**. Proceedings of the 2004 International Workshop on Hot Topics in Peer-to-Peer Systems. 2004.

ROWSTRON Antony and DRUSCHEL Peter. **Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems.** Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg. 2001.

GOTZ Stefan, RIECHE Simon, WEHRLE Klaus. **Selected DHT Algorithms.** Book Chapter in Peer-to-Peer Systems and Applications 2005.

PENDER, Tom. **UML, a Bíblia.** (LIVRO) 2 reimp. Rio de Janeiro: Campus, 2004.

PASQUINI Rafael, VERDI Fábio L., OLIVEIRA Rodolfo, MAGALHÃES Maurício F. and WELIN Annikki. **A Proposal for an XOR-based Flat Routing Mechanism in Internet-like Topologies.** In IEEE Globecom 2010. 2010-b.