Dissertação de Mestrado

Inatel
Instituto Nacional de Telecomunicações

Genesis System: Proposta de uma Arquitetura Conceitual para Orquestração de Serviços em Ambientes Autonômicos Orientados a Serviço

Rodrigo Carneiro Brandão

Março 2012

Genesis System: Proposta de uma Arquitetura Conceitual para Orquestração de Serviços em Ambientes Autonômicos Orientados a Serviço.

RODRIGO CARNEIRO BRANDÃO

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.
Coorientador: PROF. DR. LUCIANO LEONEL MENDES

Brandão, Rodrigo Carneiro
B819g

Genesis System: Proposta de uma Arquitetura Conceitual para Orquestração de Serviços em Ambientes Autonômicos Orientados a Serviço. / Rodrigo Carneiro Brandão. – Santa Rita do Sapucaí, 2012.

136 p.

Orientadores: Dr. Antônio Marcos Alberti; Dr. Luciano Leonel Mendes.

Dissertação de Mestrado – Engenharia de Telecomunicações – Instituto Nacional de Telecomunicações – INATEL.

Inclui bibliografia.

1. Serviços 2. Orquestração 3. Computação orientada a serviços 4. Composição semântica 5. Entidades habitantes 6. Engenharia de Telecomunicações I. Alberti, Antônio Marcos II. Mendes, Luciano Leonel III. Instituto Nacional de Telecomunicações – INATEL. IV. Título.

CDU 621.39

Dissertação defendida e aprovada em 22/03/2012, pela comissão julgadora:

_____

Prof. Dr. Antônio Marcos Alberti (Orientador) – INATEL

_____

Prof. Dr. Luciano Leonel Mendes (Coorientador) – INATEL

_____

Prof. Dr. Antônio Alfredo Ferreira Loureiro – UFMG

_____

Prof. Dr. Carlos Alberto Ynoguti – INATEL

_____

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

Dedico este trabalho a meus pais e a meu irmão.

"Não é o mais forte que sobrevive, nem o mais inteligente, mas o que melhor se adapta às mudanças."

Charles Darwin

Agradecimentos

Primeiramente, agradeço a Deus e a Nossa Senhora por me concederem saúde, paz e perseverança para alcançar meus objetivos e vencer mais uma etapa da minha vida acadêmica.

A minha mãe Teolinda Maria Carneiro Brandão, a meu pai Waldir Pires Brandão e a meu irmão Rômulo Carneiro Brandão, pelo apoio, incentivo, compreensão e amor que foram essenciais para superar os obstáculos enfrentados durante o curso de mestrado.

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, pela troca de ideias, apoio e por sempre apontar a direção que eu devia seguir. Ao meu coorientador Dr. Luciano Leonel Mendes, pela compreensão, atenção e palavras de incentivo.

Aos meus amigos e colegas de mestrado pelos momentos de diversão, ajuda e companheirismo durante estes dois anos.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

ACE	*Autonomic Communication Element*
ACK	*Acknowledge*
AP	*Access Point*
BIONETS	*BIOlogically-inspired NETworks and Services*
CASCADAS	*Component-ware for Autonomic Situation-aware Communications, and Dynamically Adaptable Services*
CDS	*Compilation and Decompilation System*
CID	*Content Identifier*
CISE	*Computer and Information Science and Engineering*
CNQ	*Cross Network Query*
CO	*Contract Opportunity*
DHT	*Distributed Hash Table*
DHTS	*Distributed Hash Table System*
DIET	*Decentralised Information Ecosystem Technologies*
DS	*Domain System*
EXE	*Executable*
FIA	*Future Internet Architecture*
FP6	*Sixth Framework Programme*
GA	*Goal Achievable*
GIRS	*Generic Indirection Resolution System*
GN	*Goal Needed /*
GS	*Genesis System*
HID	*Host Identifier*
ID	*Identifier*
IoS	*Internet of Service*
IP	*Internet Protocol*
KA	*Knowledge Atom*
KC	*Knowledge Container*
KN	*Knowledge Network*
MP3	*MPEG 1 Layer-3*
MPEG	*Moving Picture Experts Group*
NACK	*Negative-acknowledge*
NFC	*Near Field Communication*
NRS	*Name Resolution System*
NSF	*National Science Foundation*
OBS	*Orchestration Broker System*

P2P	*Peer-to-peer*
PDF	*Portable Document Format*
PSS	*Public Subscribe System*
QoE	*Quality of Experience*
QoS	*Quality of Service*
RFID	*Radio Frequency Identification*
ROI	*Return on Investment*
RS	*Reputation System*
S	*Service*
S_C	*Service Computation*
SDS	*Search and Discovery System*
SDS	*Semantic Data Space*
SE	*Search Engine*
S_I	*Service in Information*
SID	*Service Identifier*
SLA	*Service Level Agreements*
SOA	*Service Oriented Architecture*
SOC	*Service-Oriented Computing*
TI	*Tecnologia de Informação*
TTL	*Time To Live*
U	*User*
UML	*Unified Modeling Language*
URL	*Uniform Resource Locator*
WPA 2	*Wi-Fi Protected Access 2*
WSAN	*Wireless Sensor and Actuator Network*
XIA	*eXpressive Internet Architecture*
XID	*eXpressive Identifier*
XIP	*eXpressive Internet Protocol*
XML	*eXtensible Markup Language*

Resumo

Neste trabalho são abordadas as principais definições e características de uma Internet de serviços, conceito no qual “tudo é visto como serviço”. Aborda ainda, algumas propostas de redes futuras centradas em serviços e de núcleo misto, centradas em *hosts*, serviços e conteúdos. Com base nestes estudos, é proposto o Sistema Genesis, uma arquitetura conceitual caracterizada pelo fraco acoplamento, reusabilidade e composição de serviços. Seu principal benefício refere-se à flexibilidade intrínseca adicionada a serviços que podem ser orquestrados em tempo de execução atendendo de forma ágil, adequada e satisfatória às necessidades de processos de negócios específicos, complexos e dinâmicos.

Palavras chave: Serviços, orquestração, computação orientada a serviços, arquitetura orientada a serviços, composição semântica, entidades habitantes, oportunidades de contrato, ciclo de vida, Internet do futuro.

Abstract

This work discusses the main concepts and features of an Internet of Services, concept in which “everything is seen as a service”. Also discusses some proposals for future network centric-services and mixed kernel, centered on hosts, services and content. Based on these studies, we propose the Genesis System, a conceptual architecture characterized by loose coupling, reusability and composition of services. Its main benefit refers to the intrinsic flexibility of the added services that can be orchestrated at runtime view of a flexible, appropriate and satisfying the needs of specific business processes, complex and dynamic.

Keywords: Services, orchestration, service-oriented computing, service-oriented architecture, semantic composition, entities inhabitants, contract opportunity, life cycle, future Internet.

Capítulo 1

Introdução

1.1 Contextualização e Motivação

Desde o surgimento da Internet, esta tem crescido e se popularizado rapidamente. Em 2000 a Internet contava com aproximadamente 361 milhões de usuários e em 2011 com pouco mais de 2,1 bilhões de usuários que correspondem a aproximadamente 30,5% da população mundial [INTERNET WORLD STATS, 2011].

Este crescimento se deve ao fato de que o modelo da Internet atual foi projetado utilizando o paradigma *host-centric,* sendo o núcleo da arquitetura composto por terminais identificados com endereço IP (*Internet Protocol*). Este modelo facilitou e acelerou o crescimento da Internet, uma vez que não é necessário modificar o núcleo da rede para a criação de novas aplicações. Em contrapartida, esta simplicidade praticamente "engessa" a arquitetura, tornando difícil resolver problemas estruturais como escalabilidade, mobilidade, flexibilidade, confiabilidade e gerenciamento. Habitualmente, problemas deste tipo contam com soluções paliativas que são "remendadas" à arquitetura atual, dificultando ou até mesmo impossibilitando soluções mais significativas.

A Internet afetou e tem afetado a sociedade de forma significativa, provocando mudanças na forma como as pessoas trabalham, estudam e se divertem. Mudanças que podem ser percebidas quando observamos a grande variedade de serviços oferecidos, tais como vídeos interativos, redes sociais, jogos 3D,

transmissão de voz e TV sobre redes IP denominados *Voice over IP* (VoIP) e *Television over IP* (TVoIP), entre muitos outros.

É perceptível a transição de uma economia baseada em produtos para uma economia baseada em serviços. Todos os países desenvolvidos apresentam o setor de serviços como um setor em evidência, seja na geração de empregos, atuando como diferencial competitivo ou suporte às atividades de manufatura.

A tendência é que a Internet atenda as demandas desta economia, exercendo um papel cada vez mais decisivo em todos os processos de negócio e se torne uma ferramenta de produtividade por excelência [REDING, 2008]. No entanto, o modelo atual não é flexível a ponto de permitir a criação e adaptação de serviços para atender os processos de negócios de forma satisfatória, rentável e com baixo custo.

Além das dificuldades deparadas, a utilização da Internet pelos usuários está relacionada em sua maioria a processos de descoberta, negociação, contratação e utilização de serviços e conteúdos. Fatores que contribuem para que outras propostas com abordagens projetadas a partir do zero e que não são obrigatoriamente acopladas às redes IP atuais ganhem força. Estas arquiteturas são baseadas na abordagem *Clean Slate.*

A abordagem *Clean Slate* para a Internet é o mesmo que "reinventar a Internet". Esta "Nova Internet" deverá superar as limitações da Internet atual, incorporar novas tecnologias, permitir a criação de novas classes de serviços e aplicações, bem como, ser uma plataforma que suporte inovações e atenda as demandas atuais e futuras.

Em tempos de mudanças rápidas e um ambiente de negócios extremamente competitivo é preciso desenvolver alternativas que possibilitem prover serviços adequados e adaptativos. Desta forma, este trabalho agrega soluções que possibilitem modelar e representar os serviços através da Internet.

A Internet de Serviços tem como objetivo pesquisar e desenvolver novas teorias, modelos e tecnologias para prover soluções eficientes e eficazes que

permitam a todo tipo de usuário comercializar e consumir serviços disponíveis na Internet [CARDOSO, et. al., 2011]. Por meio da Internet de Serviços as organizações e usuários poderão facilmente encontrar sistemas expostos como serviços, combiná-los e adaptá-los a seu contexto específico.

A estrutura orientada a serviços remete ao paradigma de Computação Orientada a Serviços, que utiliza os serviços como elemento fundamental para o desenvolvimento de aplicações de forma rápida, com baixo custo e em ambientes distribuídos [PAPAZOGLOU, et. al., 2006].

A Arquitetura Orientada a Serviços utiliza esta computação para desenvolver sistemas e aplicações com base em serviços independentes, reutilizáveis e fracamente acoplados, a fim de orquestrar serviços que sejam capazes de atender processos de negócios específicos e se adaptar a um mercado de negócios dinâmico, complexo e competitivo.

Esta dissertação também visa clarear a relação entre Internet do Futuro e TV digital. O INATEL participou do processo de definição do padrão brasileiro de TV digital e possui diversos trabalhos nesta área. Assim, para dar continuidade a estes trabalhos, torna-se necessário entender melhor a relação entre estas tecnologias. O objetivo é determinar como ambas as tecnologias podem ser integradas, como a TV digital pode tirar proveito dos avanços tecnológicos provenientes das pesquisas em Internet do Futuro e como arquiteturas de redes centradas em informação e serviços podem tirar proveito dos sistemas de TV digital.

1.2 Objetivo da Dissertação

O objetivo desta dissertação é realizar o estudo e análise das principais características de arquiteturas de serviços e com base nos requisitos e desafios levantados na fase de análise, propor o Sistema Genesis, uma arquitetura de rede conceitual que possibilita orquestrar serviços semanticamente, isto é, baseado no significado aos usuários. Esta proposta contempla ainda, a apresentação dos sistemas necessários para o bom funcionamento da arquitetura conceitual. Por fim, é proposto um modelo para o ciclo de vida de serviços.

1.3 Organização da Dissertação

O restante desta dissertação é organizado como segue: O capítulo 2 apresenta o paradigma de Computação Orientada a Serviço, as características, propósitos e sua implementação através da Arquitetura Orientada a Serviços. Para melhor compreensão, são apresentados ainda neste capítulo, os principais conceitos e propriedades associadas ao termo "serviço", bem como, o ciclo de vida e os benefícios da arquitetura. O capítulo 3 aborda o estudo de três propostas de arquitetura para ambientes orientados a serviços. No capítulo 4 são abordados os sistemas básicos da arquitetura proposta e a ampliação do Sistema Generalizado de Resolução de Indireções proposto por [MARTINS, 2011] na direção de uma arquitetura convergente para a Internet de Informações e Serviços. O capítulo 5 contém a proposta principal desta dissertação, o Sistema Genesis, responsável pela instanciação e criação dos demais sistemas e entidades da arquitetura. Este capítulo aborda ainda, três estudos de caso para exemplificar as formas de localização e contratação de serviços em ambientes altamente dinâmicos e distribuídos. O capítulo 6 contém a conclusão desta dissertação, bem como algumas propostas de trabalhos futuros.

Capítulo 2

Ambientes Orientados a Serviços

2.1 Serviços: Conceitos e Propriedades

Um serviço é um recurso abstrato que representa uma capacidade de realizar tarefas específicas. É normalmente executado quando necessário, mas permanece inativo até que seja solicitado. Novos serviços podem ser oferecidos através da combinação de serviços existentes [DUBRAY, 2005]. Os correios, por exemplo, podem utilizar o serviço de transporte oferecido por outras empresas para enviar documentos e encomendas entre um remetente e um destinatário.

Para melhor entendimento do termo "serviços", serão abordados nas subseções 2.1.1 e 2.1.2 os principais conceitos e propriedades relacionadas.

2.1.1 Conceito de Serviço

Os termos mais relevantes associados com o conceito de serviços e introduzidos ao longo do tempo pela comunidade científica e industrial [CARDOSO, et. al., 2011] são: Serviços do Mundo Real, *E-Service*, *Web Service* e Internet de Serviços. A **figura 2.1** redesenhada de [CARDOSO, et. al., 2011] ilustra a trajetória destes serviços.

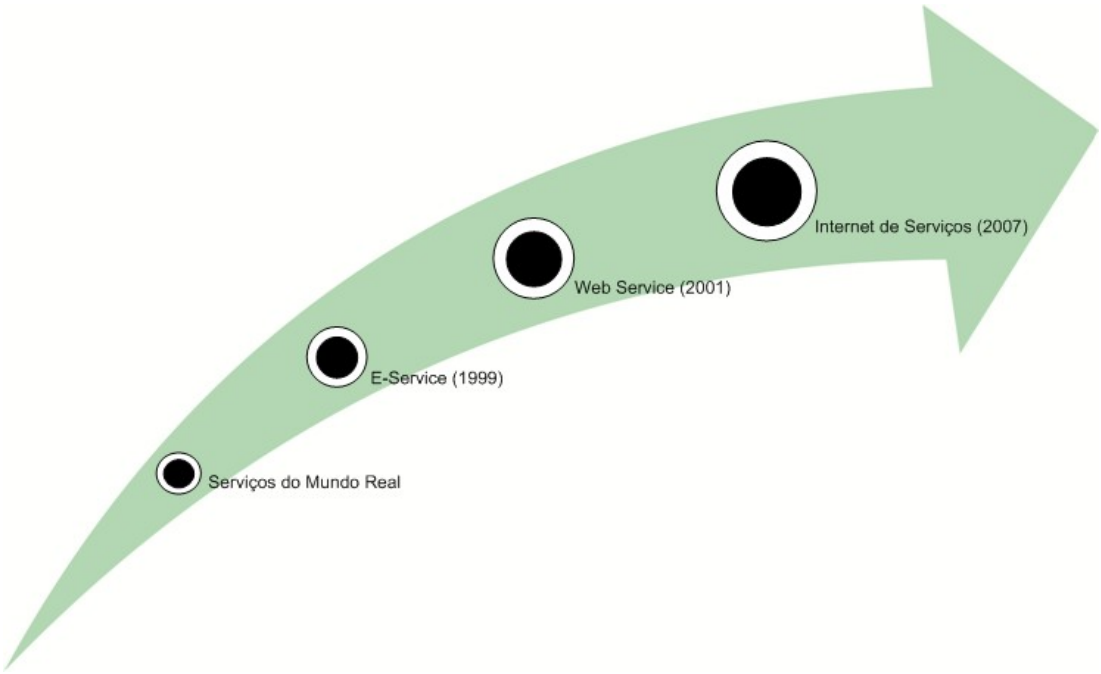

Figura 2.1: *Termos associados com o conceito de serviços [CARDOSO, et. al., 2011].*

Serviços do Mundo Real: É o termo usado para se referir a qualquer tipo de serviço que pode ser encontrado na sociedade. Devido à sua diversidade e heterogeneidade, tem sido tradicionalmente difícil definir seu significado. Serviço pode ser definido como qualquer atividade ou benefício que uma parte possa oferecer a outra, e que seja basicamente intangível, podendo sua produção estar ou não vinculada a um produto físico [KOTLER, 1998]. Por exemplo, qualquer pessoa que exerça uma tarefa distinta no apoio a outra, está provendo um serviço. Da mesma forma, uma organização que realiza tarefas associadas com seu negócio, também está provendo um serviço.

***E-Service*:** O termo "*E-Service"* (abreviação de *Electronic Service*) é usado para definir serviços que utilizam redes de dados como um canal de comunicação, permitindo a interação entre consumidores e serviços remotos. Praticamente, qualquer serviço pode ser transformado em um *e-service,* desde que possa ser invocado através de uma rede de dados. *E-services* são independentes da linguagem de especificação usada para definir a sua funcionalidade, interface ou propriedades não funcionais, e, tal como acontece com os serviços do mundo real, sua definição é bastante ampla [CARDOSO, et. al., 2011].

Web Service: Permite que diferentes aplicativos de *software* se comuniquem facilmente, independente da linguagem de programação ou plataforma de computação. Esta tecnologia possibilita que novas aplicações possam interagir com aquelas que já existem e que sistemas desenvolvidos em plataformas diferentes sejam compatíveis. Cada aplicação pode ter a sua própria linguagem de programação, que é traduzida em uma linguagem universal, o formato XML (*Extensible Markup Language*). O *Web service* é o componente que permite às aplicações enviar e receber dados neste formato.

Internet de Serviços: É o termo designado para identificar os serviços oferecidos através da Internet.

Duas propriedades diferenciam os serviços baseados na Internet de Serviços (*IoS - Internet of Services*) dos demais tipos de serviços já apresentados [CARDOSO, et. al., 2011]: a primeira é que a noção de serviço não se limita apenas aos serviços relacionados à Tecnologia de Informação (TI), mas também aos serviços do mundo real. E, a segunda, é que as partes interessadas no fornecimento e consumo de serviços, não se restringe apenas aos profissionais de TI, mas a todos os usuários da rede. Desta forma, os serviços baseados na IoS podem ser utilizados diretamente pelos consumidores, e, invocados por sistemas técnicos para acessar funcionalidades de negócio oferecidas remotamente por provedores de serviços.

2.1.2 Propriedades Intrínsecas do Serviço

Tão importante quanto definir o conceito de serviços, é conhecer as suas propriedades intrínsecas e as implicações na gestão de operações. Antes de propor ou construir uma solução baseada em serviços, é fundamental compreender a natureza dos serviços do mundo real, uma vez que serão digitalizados e representados adequadamente para serem então comercializados na nova Internet. A **figura 2.2** ilustra as características intrínsecas dos serviços do mundo real [CARDOSO, et. al., 2011].

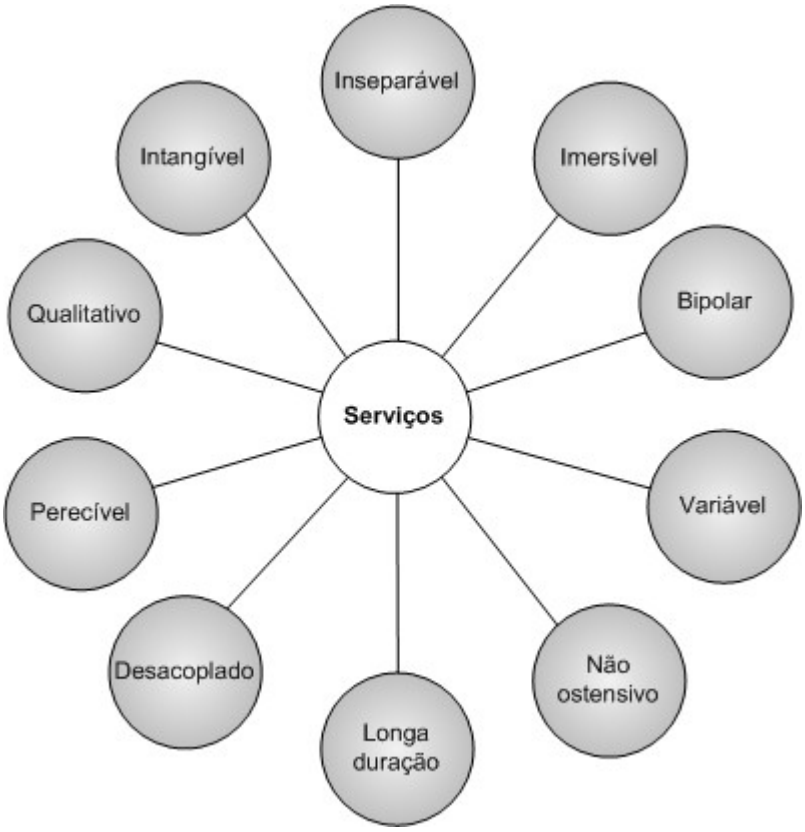

Figura 2.2: *Características Intrínsecas dos Serviços.*

Intangível: Que não pode ser tocado (palpável). Serviços são intangíveis, uma vez que não existem materialmente. Isto é, não podem ser tocados ou possuídos pelo consumidor como se fossem bens manufaturados, mas apenas utilizados. Muitas vezes é possível ver e avaliar os resultados que surgem a partir da execução de um serviço, mas não tocá-lo como se fosse um produto material [CARDOSO, et. al., 2011]. Por exemplo, não é possível tocar uma consulta médica ou um suporte especializado em TI.

Inseparável: Que não pode ser separado. O provimento e o consumo de serviços ocorrem geralmente em paralelo, sendo o cliente, o responsável pelo início do processo. É importante que haja um severo equilíbrio de oferta e demanda, caso contrário, os serviços serão perdidos ou os consumidores terão que aguardar a disponibilidade do serviço [CARDOSO, et. al., 2011].

Imersivo: Serviços são normalmente executados com base na colaboração de seus provedores e consumidores, sendo em muitos casos, difícil determinar os responsáveis pelo seu sucesso ou fracasso. Desta forma, é de extrema

importância estabelecer Acordos de Nível de Serviço (SLAs - *Service Level Agreements*) que abordem os padrões, funções e processos definidos e aceitos por ambas as partes [CARDOSO, et. al., 2011], delineando responsabilidades e estabelecendo penalidades em caso de níveis insatisfatórios na prestação dos serviços contratados.

Bipolar ou híbrido: Denota a atuação de dois mecanismos distintos. A execução de serviços muitas vezes envolve a combinação de recursos humanos e tecnológicos. Embora as abordagens para monitorar as complexas relações entre as dimensões humanas e tecnológicas não tenham sido estudadas no contexto de ambientes altamente distribuídos, autônomos e heterogêneos como a Internet de Serviços [CARDOSO, et. al., 2011].

Variável: Que pode apresentar diferentes aspectos. Ao contrário dos produtos que têm alto grau de padronização, os serviços muitas vezes são baseados no cliente, o que impossibilita uma produção totalmente uniforme. A qualidade e consistência dos serviços estão sujeitos a uma considerável variabilidade, uma vez que dependem dos seres humanos e suas atitudes (habilidades cognitivas, emoções, perfis e comportamentos, por exemplo) [CARDOSO, et. al., 2011].

Não Ostensivo: Que não se pode mostrar ou exibir. A maior parte dos serviços, se não todos, não pode ser ostentada. Não é possível possuir ou exibir um serviço.

Longa duração: A comercialização de produtos requer baixo nível de interação entre os fornecedores e consumidores. Já os serviços requerem um maior nível de interação entre os provedores e consumidores. Este é um processo longo, que passa por etapas, como: localização, negociação, contratação e prestação do serviço [CARDOSO, et. al., 2011].

Desacoplado: Que é separado, independente. O ciclo de vida de um serviço inclui basicamente cinco fases: descoberta, seleção, invocação, execução e rescisão. Essas fases podem ser realizadas manualmente por seres humanos com o auxílio de tecnologias ou de forma automática. É importante ressaltar que na IoS

cada fase é desacoplada, sendo realizadas de forma independente [CARDOSO, et. al., 2011].

Perecível: Que pode se extinguir. Devido os serviços serem intangíveis, eles não podem ser armazenados. Como consequência, os serviços são perecíveis, não sendo possível armazenar sua capacidade inutilizada para negociações futuras.

Qualitativo: Relativo à qualidade, à natureza dos objetos, mas não à quantidade. A prova física ou os produtos tangíveis resultantes da execução de serviços podem fornecer indicações que permitem medir sua qualidade [CARDOSO, et. al., 2011].

Todas as propriedades descritas, precisam ser exploradas nos serviços do mundo real e modeladas digitalmente nas propostas para IoS, possibilitando que os clientes utilizem serviços remotamente através da nova Internet.

2.2 Computação Orientada a Serviços

A Computação Orientada a Serviços (SOC – *Service-Oriented Computing*) é o paradigma de computação que utiliza serviços como o elemento básico para o desenvolvimento e composição de aplicações distribuídas, mesmo em ambientes heterogêneos. Serviços são autônomos, independentes de plataforma computacional, fracamente acoplados, podendo ser descritos, publicados, descobertos e dinamicamente combinados. A criação de serviços reflete uma programação orientada a serviços, baseada na idéia de compor aplicações através da descoberta e invocação de serviços disponíveis na rede, ao invés de desenvolver ou utilizar aplicações disponíveis para realizar alguma tarefa específica [PAPAZOGLOU, et. al., 2006].

Os serviços podem ser oferecidos por diferentes provedores, se comunicar através da Internet e serem utilizados por clientes para atender desde tarefas simples a complexos processos de negócios. Para que seja possível entender exatamente o que isso significa, a **figura 2.3** ilustra os principais princípios e objetivos da computação orientada a serviços.

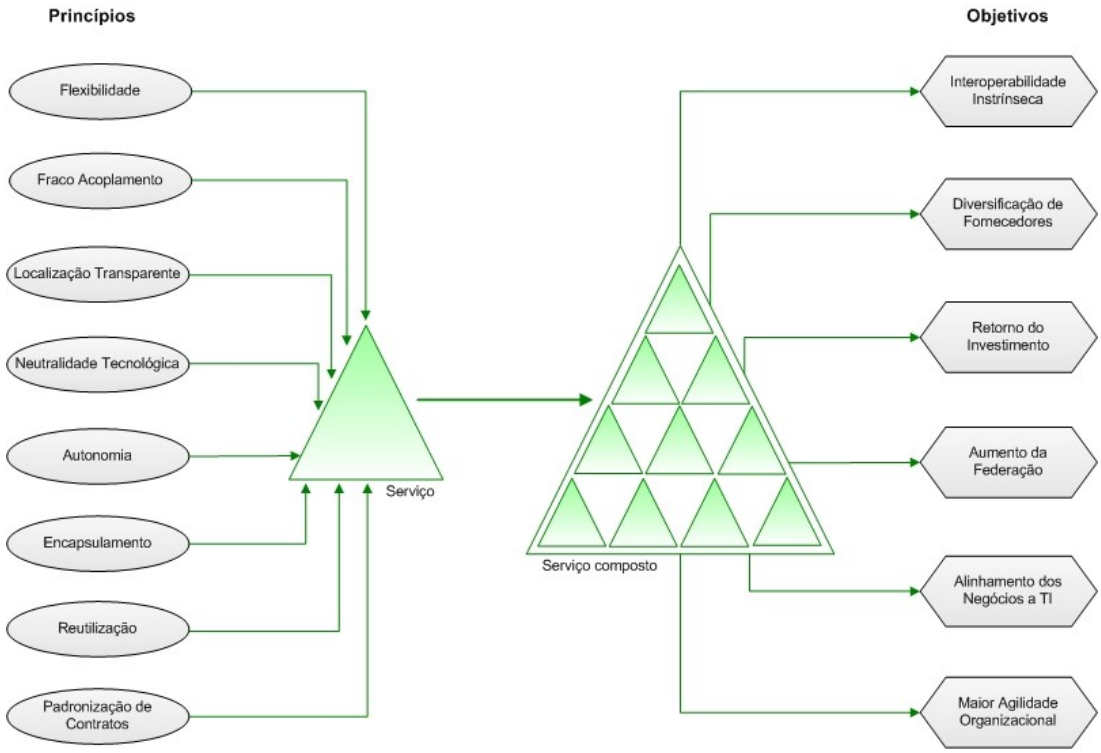

Figura 2.3: *Princípios e objetivos do paradigma de computação orientada a serviços.*

Conforme ilustrado na **figura 2.3**, os principais princípios da computação orientada a serviços são:

Flexibilidade: O cenário de negócios tradicional exige que as empresas alterem continuamente seus sistemas para atender as mudanças do mercado, como por exemplo, novas exigências e requisitos [CARTER, 2007]. SOC permite um melhor alinhamento entre a TI e os negócios, com a criação de novos serviços e a reutilização dos serviços já existentes, a fim de atender de forma eficiente e eficaz as mudanças do mercado [ENDREI, et. al., 2004].

Fraco Acoplamento: Os serviços são fracamente acoplados por natureza. Isto é, são executados em diferentes plataformas, implementados de forma independependente e possuem diferentes proprietários [AIELLO, et. al., 2006]. O acoplamento fraco libera os clientes de qualquer conhecimento da estrutura interna dos serviços, e, os clientes não devem exigir que o serviço conheça o contexto que será utilizado [PAPAZOGLOU, et. al., 2006]. A idéia de serviços fracamente acoplados é evitar que as modificações, falhas ou até mesmo exclusões de serviços

causem conflitos a outros serviços, devendo estes ser tolerantes a falhas, flexíveis e escaláveis.

Localização Transparente: Os serviços devem ter as suas definições e informações de localização armazenadas em um repositório que seja acessível a uma variedade de clientes, possibilitando que o serviço seja localizado e invocado, independentemente de onde será executado [PAPAZOGLOU, et. al., 2006]. Esta habilidade permite clientes e provedores de serviços agirem independentes de suas localizações, podendo, por exemplo, um cliente descobrir e utilizar um serviço sem saber onde seu provedor está localizado.

Neutralidade Tecnológica: Serviços devem ser invocados por meio de tecnologias padronizadas, disponíveis em praticamente todos os ambientes de TI. Isto implica que os mecanismos de invocação dos serviços, como descritores de serviços e mecanismos de descoberta, estejam de acordo com os padrões aceitos [PAPAZOGLOU, et. al., 2006].

Autonomia: Os serviços devem ser implantados, modificados e mantidos de forma independente um do outro, sendo o ciclo de vida de cada serviço independente do ciclo de vida do outro [ROSEN, et. al., 2008].

Encapsulamento: Os detalhes da implementação interna dos serviços e sua estrutura de dados deve estar "escondida". Deve haver uma separação rigorosa da interface do serviço (o que ele faz) da sua implementação (como é feito) [ROSEN, et. al., 2008].

Reutilização: Permite que os serviços existentes sejam compartilhados e reutilizados para a criação de aplicações ou serviços compostos. Desta forma, os provedores e consumidores de serviços desfrutam de vantagens, como a redução no custo de investimento, personalização do *software*, e a redução do tempo que leva desde a concepção de uma aplicação até sua chegada ao mercado.

Padronização de Contratos: Os serviços expressam o seu propósito e capacidades de contratos formais. Isto é, após as etapas de negociação (apresentadas na seção 2.4), faz-se necessário estabelecer um contrato que formalize o

entendimento e expectativas das partes envolvidas (consumidor e provedor de serviços). A padronização de contratos é importante para assegurar que os parâmetros estabelecidos pelos serviços sejam coerentes, confiáveis e gerenciáveis [SOA PRINCIPLES, 2012].

SOC propõe uma grande mudança de paradigma na maneira que as aplicações de *software* são concebidas, arquitetadas, disponibilizadas e consumidas. Apesar das dificuldades inerentes a todo tipo de mudança, a **figura 2.3** ilustra ainda, os principais objetivos da computação orientada a serviços.

Aumento da Interoperabilidade Intrínseca: Consiste em instituir a interoperabilidade nativa dentro dos serviços, a fim de reduzir a necessidade de integração. Isto possibilita que diferentes serviços possam ser repetidamente agrupados em uma variedade de composições para realizar uma série de processos de negócios [SOA PRINCIPLES, 2012].

Diversificação de Fornecedores: Diversificação refere-se à capacidade que o cliente tem de analisar diversos fornecedores e serviços, invocando o melhor serviço ou aquele que seja mais condizente com sua demanda. O cliente poderá diversificar o fornecedor sempre que julgar necessário.

Retorno do Investimento (*ROI - Return on Investment*): Mensurar o retorno do investimento de soluções automatizadas é uma questão crítica na determinação de quão rentável uma determinada aplicação ou sistema realmente é. Para muitas organizações, a sobrecarga financeira requerida pela TI é uma questão preocupante, uma vez que não contribui ativamente para aumentar o valor do negócio. Neste sentido, computação orientada a serviços defende a criação de serviços que possam ser reaproveitados diversas vezes para automatizar e se adequar a diferentes processos de negócios, como parte de diferentes soluções [SOA PRINCIPLES, 2012].

Aumento da Federação: Um ambiente de TI federado é aquele em que os recursos e aplicações estão unidos mantendo a sua autonomia individual e

autogestão. Ambientes orientados a serviços ampliam esta federação através da implantação de serviços padronizados e combináveis [SOA PRINCIPLES, 2012].

Alinhamento dos Negócios a TI: O cenário de negócios tradicional exige que as empresas alterem continuamente seus sistemas para atender as mudanças do mercado, como por exemplo, novas exigências e requisitos [CARTER, 2007]. SOC permite um melhor alinhamento entre a TI e os negócios, com a criação de novos serviços e a reutilização dos serviços já existentes, a fim de atender de forma eficiente e eficaz as mudanças do mercado [ENDREI, et. al., 2004].

Maior Agilidade Organizacional: Ser capaz de adaptar-se rapidamente às mudanças da indústria e superar a estratégica da concorrência é um fator atraente para as empresas, especialmente as do setor privado. SOC possibilita esta adaptação por meio da criação de serviços padronizados e reutilizáveis e, portanto, independentes de processos de negócios [SOA PRINCIPLES, 2012].

2.3 Arquitetura Orientada a Serviços

A Arquitetura Orientada a Serviços (SOA – *Service-Oriented Architecture*) é um estilo de arquitetura de *software* cujo princípio fundamental é que as funcionalidades implementadas pelas aplicações sejam criadas a partir de serviços independentes com interfaces bem definidas, que podem ser invocadas em sequências definidas para formar processos de negócio [CHANNABASAVAIAH, et. al., 2004], possibilitando fornecer serviços para usuários finais, usuários especialistas e serviços distribuídos em rede.

Para se construir uma SOA é preciso ter uma compreensão clara dos processos e domínios de negócios funcionais, incluindo decisões de como implementar funções de negócios como serviços separados. A modelagem e concepção de serviços dentro desta arquitetura requer abordagens mais sofisticadas que as utilizadas em aplicações tradicionais [GÜNER, 2005].

A arquitetura básica do SOA envolve o relacionamento entre três participantes: o Provedor do Serviço, o Consumidor do Serviço e o Serviço de Registro, conforme ilustrado na **figura 2.4** redesenhada de [ENDREI, 2004].

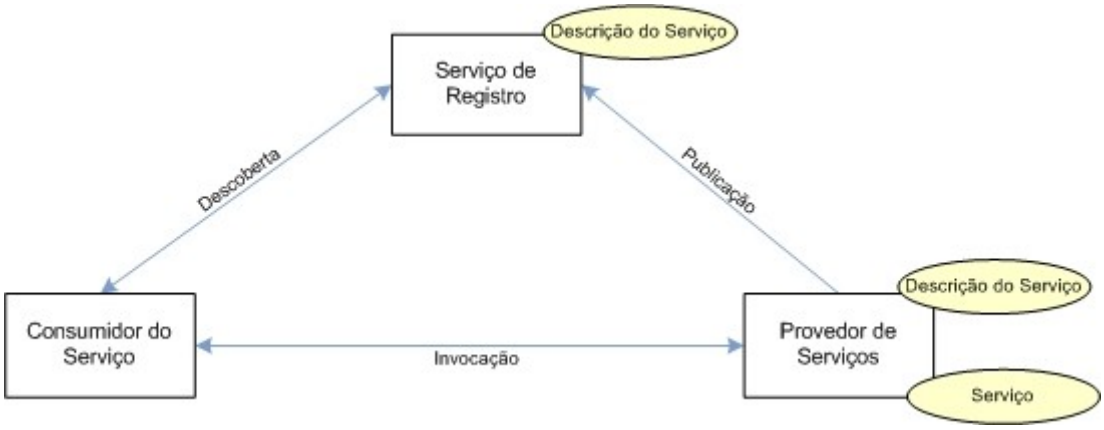

Figura 2.4: *Arquitetura básica do SOA [ENDREI, 2004].*

O Consumidor do Serviço ou Cliente pode ser uma aplicação, um módulo de *software* ou outro serviço que esteja requerendo um serviço específico. O Provedor de Serviços é a entidade que disponibiliza um serviço, recebe e processa as solicitações de pedidos enviadas pelos Clientes; e por fim, o Serviço de Registro ou *Broker* é a entidade responsável por manter as publicações das descrições de serviços, facilitando sua descoberta por parte dos Consumidores.

Conforme ilustrado na **figura 2.4**, o relacionamento entre os três participantes envolve basicamente as operações de publicação, descoberta e invocação. A publicação ocorre quando a descrição de um serviço é publicada junto ao Serviço de Registro, possibilitando que os Clientes possam descobrir e invocar os serviços de seu interesse. Descoberta acontece quando um solicitante de serviços localiza um serviço por meio do Serviço de Registro. Por fim, a invocação ocorre quando o Consumidor invoca o serviço cuja descrição tenha lhe interessado.

2.3.1 Ciclo de Vida SOA

O ciclo de vida representa a forma como os serviços são criados para atender a processos de negócio dinâmicos e específicos. A **figura 2.5** ilustra o ciclo de vida SOA, que segundo [LALIWALA, 2007] contempla quatro fases: Modelagem, Composição, Implantação, e por fim, Execução e Monitoramento.

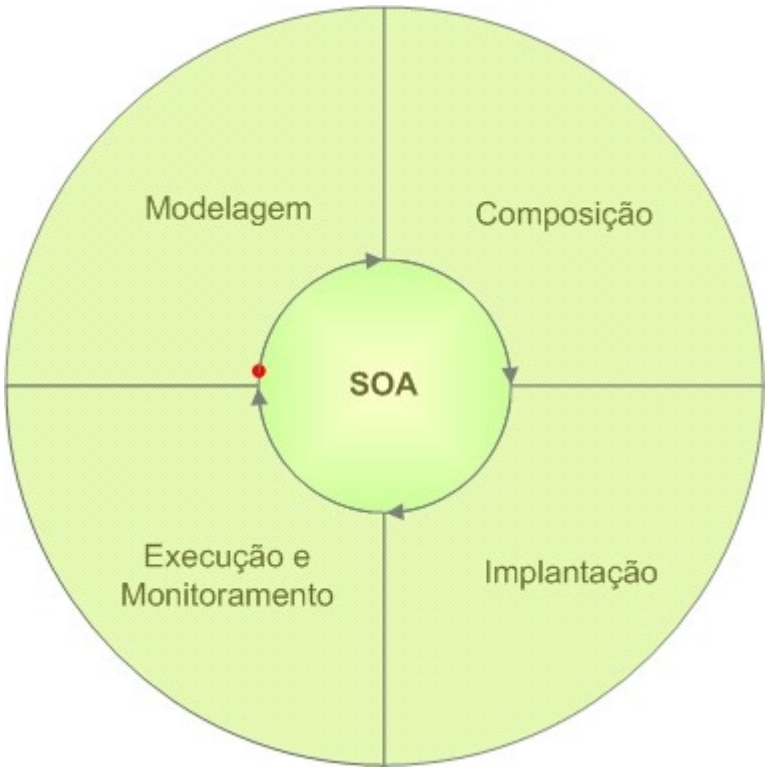

Figura 2.5: *Ciclo de Vida de Eventos da Arquitetura Orientada a Serviços.*

Modelagem: Inclui a análise e desenho dos processos de negócios, serviços, eventos e mensagens. Nesta fase, os serviços são modelados para executar uma função específica, podendo vir a se tornar parte de um serviço mais complexo, um serviço composto. Ainda nesta fase, é importante definir uma ontologia que forneça um vocabulário de domínio comum e conhecimento dos modelos de serviços de negócios [LALIWALA, 2007].

Composição: A composição de serviços compreende as funcionalidades para criar ou reutilizar serviços já existentes (simples e compostos) e combiná-los. Não apenas os serviços simples, mas também os compostos podem ser utilizados como serviços básicos na composição de outros serviços [PAPAZOGLOU, et. al., 2007]. Antes do fornecimento de serviços, deve existir uma etapa de negociação e ser estabelecido um SLA entre as partes envolvidas [CARDOSO, et. al., 2011].

Implantação: Envolve a criação de ambientes para a implantação e provisionamento de serviços. Os serviços devem ser implantados de forma integrada

e executados em uma sequência que possibilite atender um processo de negócio composto [LALIWALA, 2007].

Execução e Monitoramento: O objetivo desta fase é permitir a execução do serviço conforme os requisitos acordados e observar seu desempenho, garantindo a qualidade na execução e visando disparar eventos de adaptabilidade caso sejam necessários.

2.3.2 Benefícios SOA

SOA proporciona serviços ágeis, adaptativos e flexíveis, e, uma maior eficiência na manutenção e evolução de sistemas, fatores preponderantes para ambientes de negócios competitivos e que mudam rapidamente. Os benefícios proporcionados pela adoção da arquitetura orientada a serviços, evidenciados por [MODI, 2004], são:

Aplicações mais produtivas e flexíveis: Permite que as empresas utilizem as vantagens das tecnologias existentes, integrando sistemas e aplicações legadas, sem a necessidade de soluções proprietárias.

Aplicações seguras e gerenciáveis: SOA fornece uma infraestrutura comum para o desenvolvimento seguro, monitorado e previsível de serviços ou aplicações, facilitando a adição de novos serviços e capacidades que mapeiam processos críticos do negócio.

Desenvolvimento de aplicações de forma rápida e rentável: Permite criar um repositório de serviços reutilizáveis que podem ser combinados e como resultado, compor uma aplicação que poderá atender uma necessidade de negócio específico.

Melhoria no processo de tomada de decisão: Ao agregar aplicações e informações de negócios em um conjunto de serviços dinâmicos, é possível tomar decisões mais precisas e obter informações mais completas.

A implantação de uma estratégia SOA permite que as empresas estejam prontas para o futuro. Serviços de negócios podem ser facilmente criados, modificados e combinados para conseguir atender as necessidades da época. É importante ressaltar que o principal benefício do SOA é adquirido ao longo do tempo, por meio das mais diversas implementações e composição de novos serviços.

2.4 Negociação em Ambientes Orientados a Serviços

O modelo orientado a serviços propõe que o *software* seja desenvolvido e disponibilizado como um serviço, podendo ser entregue e utilizado sob demanda e não comercializado como um produto, como habitualmente é feito. Neste modelo, os elementos dos serviços são identificados, a prestação do serviço é negociada, o serviço é executado e então descartado. O fato de poder ser descartado é o que diferencia o "serviço" do "produto", podendo o serviço ser utilizado sob demanda e por diversos consumidores. A principal vantagem dessa abordagem é o fraco acoplamento entre as demandas de negócios e as soluções de *software*s associadas [ELFATATRY & LAYZELL, 2004].

O conceito de disponibilizar *software* como serviço é relativamente simples: "Não compre *software*, apenas use-o como e quando precisar dele" [ELFATATRY & LAYZELL, 2004]. Em um modelo orientado a serviços, o mesmo serviço pode ser oferecido com diferentes qualidades, preços ou condições de entrega, visando atender as diferentes demandas dos consumidores.

Em ambientes dinâmicos, onde as ofertas e demandas estão em constante mudança, não é interessante fixar qualquer característica do serviço, como o preço, por exemplo, devendo estas características ser negociadas no momento em que o serviço for solicitado.

2.4.1 Mercado de Serviços

Mercado é o lugar onde os vendedores e os consumidores de bens ou serviços podem se reunir com o intuito de realizar uma transação. Os elementos

básicos de um mercado (digital ou físico) são os participantes, as mercadorias e as regras para a interação [ELFATATRY & LAYZELL, 2004].

Em um ambiente orientado a serviços, o mercado é considerado um elemento fundamental. Será no mercado que os provedores, consumidores e possivelmente uma terceira entidade, como, por exemplo, representantes de entidades que não estejam aptas a representar seus interesses, oferecem, solicitam e negociam serviços. De acordo com [ELFATATRY & LAYZELL, 2004], este mercado deve conter as seguintes características:

Escala: Ambientes orientados a serviços necessitam de uma grande variedade de serviços para atender as mais diversas necessidades dos consumidores. Desta forma, o mercado é caracterizado por um número crescente de serviços, provedores de serviços e solicitantes, futuros consumidores de serviços.

Conhecimento imperfeito: Por se tratar de um mercado extenso, o conhecimento dos provedores e consumidores de serviços é limitado, não sendo preciso que estes tenham um conhecimento perfeito sobre a outra parte ou do ambiente em que o serviço está inserido. Desta forma, faz-se necessária a criação de regras de confiança. Estas podem ser criadas pelos negociadores, ou por entidades independentes, como por exemplo, outros serviços.

Tempo: Os serviços são negociados a partir de cada solicitação, devendo as negociações ser concluídas em um curto intervalo de tempo, de forma que não afete a eficiência da aplicação. [ELFATATRY & LAYZELL, 2004] ressalta que uma forma de reduzir a necessidade de negociação a cada solicitação de serviço é o estabelecimento de um contrato que inclua a possibilidade do consumidor utilizar o serviço futuramente.

Volatilidade: É uma variável que mostra a intensidade e frequência das oscilações referentes à entrada e saída de consumidores e provedores no mercado de serviços. Isto afeta a relação entre demanda, oferta e preço de forma imprevisível, tornando o mercado um ambiente caracterizado por elevado grau de incerteza. Desta

forma, os consumidores devem lidar com a possibilidade de que os serviços podem não estar disponíveis sempre que desejarem.

2.4.2 Processo de Negociação

O objetivo do processo de negociação é estabelecer um SLA entre o cliente e o provedor de serviços, a fim de viabilizar a prestação de serviço requerida pelo cliente ou oferecida pelo provedor. O processo de negociação em um ambiente orientado a serviços pode ser dividido em três fases: Pré-negociação, Negociação e Entrega do serviço. A **figura 2.6** redesenhada de [ELFATATRY & LAYZELL, 2004] ilustra estas fases.

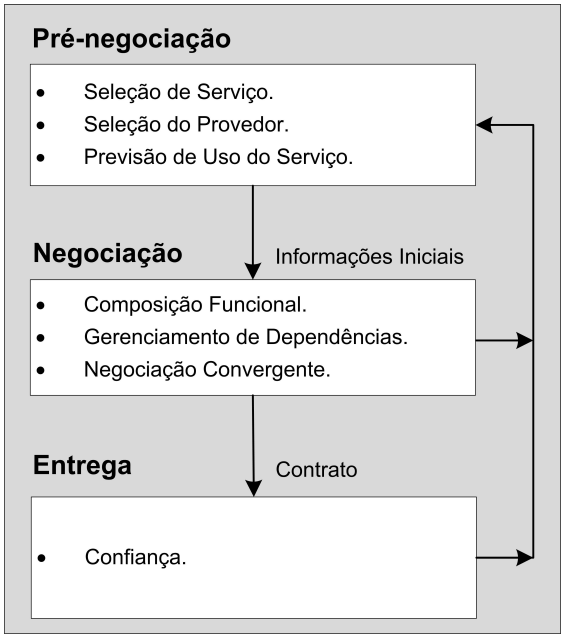

Figura 2.6: *Fases da Negociação de Serviços [ELFATATRY & LAYZELL, 2004].*

2.4.2.1 Pré-negociação

Nesta fase, o objetivo é o estudo e entendimento da negociação, devendo cada negociador planejar e elaborar a melhor alternativa para estabelecer um acordo. Esta fase envolve a seleção do serviço, seu provedor, e a previsão de uso do serviço selecionado.

Seleção do Serviço: A seleção do serviço em ambientes orientados a serviço depende dos seus atributos funcionais e não funcionais. Funcionais são

aqueles ligados diretamente ao objetivo da demanda e as funcionalidades esperadas pelo cliente. Não funcionais, são aqueles que não estão relacionados diretamente com a demanda, mas que são importantes para o desempenho do serviço, padronização de processos e satisfação do cliente de uma forma geral, como por exemplo, preço e qualidade do serviço prestado [ELFATATRY & LAYZELL, 2004].

Seleção do Provedor: Por se tratar de um mercado extenso, cada invocação de serviços pode envolver provedores distintos. Desta forma, [ELFATATRY & LAYZELL, 2004] propõem que o consumidor de serviços mantenha uma pequena lista de provedores de sua confiança e preferência, sendo esta atualizada frequentemente com base no *feedback* obtido nas fases de negociação e entrega dos serviços.

Previsão de Uso do Serviço: Em um sistema onde grande parte ou todas as funcionalidades são disponibilizadas como serviços, o grande número de negociações gera uma sobrecarga de comunicação, podendo afetar o desempenho do serviço. Desta forma, a opção proposta por [ELFATATRY & LAYZELL, 2004] é que seja possível estabelecer contratos temporários assim que o serviço é invocado pelo cliente. Considere, por exemplo, um serviço de processador de texto. Neste caso, a aplicação se comunica com provedores de serviço de verificação ortográfica e negocia os termos e condições para que tal serviço seja inicializado assim que o usuário iniciar a edição de textos. Quando o usuário utiliza o editor de textos, o serviço de verificação ortográfica é invocado e o contrato é assinado [ELFATATRY & LAYZELL, 2004].

2.4.2.2 Negociação

A negociação é iniciada no momento em que o provedor e o consumidor de serviços trocam mensagens com o objetivo de estabelecer um acordo, que posteriormente será formalizado por meio de um contrato. Cada contrato contém a descrição do serviço negociado, além de uma série de condições do negócio. A fase de negociação pode ser dividida em três etapas:

Composição Funcional: Em um mercado com vários serviços e provedores de serviços, a composição de funcionalidades é uma questão primordial e desafiadora. Diferentes blocos funcionais são fracamente acoplados. Desta forma, é importante levar em consideração os efeitos finais desta composição e qual será a qualidade da aplicação originada pela combinação de diferentes serviços [ELFATATRY & LAYZELL, 2004].

Gerenciamento de Dependências e Incertezas: Ao contrário de processos de negócios tradicionais que normalmente são executados de uma forma relativamente previsível e repetitiva, a composição de serviços lida com questões de incerteza e gestão de dependências em um ambiente dinâmico.

Negociação Convergente: Em um ambiente onde o tempo é uma questão crítica, as negociações devem ser concluídas em um curto intervalo de tempo. Dois mecanismos podem ajudar a alcançar este objetivo: protocolos de negociação e estratégias de negociação [ELFATATRY & LAYZELL, 2004].

Os protocolos de negociação são usados para alinhar o comportamento das entidades envolvidas, de modo que não haja desperdício de tempo. As estratégias de negociação usam duas abordagens para alcançar uma melhor convergência da negociação: a heurística e a argumentação. A abordagem heurística utiliza diferentes técnicas de aprendizagem que permite a uma entidade gerar ofertas de serviços, que provavelmente serão aceitas, com base no histórico de negociação da outra entidade. A abordagem de argumentação depende do uso de técnicas para persuadir a outra parte a aceitar a proposta [ELFATATRY & LAYZELL, 2004].

2.4.2.3 Entrega

Como resultado da fase de negociação é estabelecido um contrato entre as partes envolvidas. O contrato descreve a funcionalidade do serviço, seus atributos não funcionais, bem como, as pré e pós-condições do serviço e do contrato. As condições do serviço são de natureza técnica e descrevem os requisitos e consequências do uso de um serviço específico. As condições do contrato são de natureza empresarial e podem ser aplicadas a mais de um serviço [ELFATATRY &

LAYZELL, 2004], avaliando os resultados obtidos, o compromisso das partes envolvidas na negociação e a satisfação dos negociadores. Dado que o contrato foi estabelecido, a entrega ou prestação de serviços é realizada.

A confiança é um fator de destaque na prestação de serviços. Afinal, em um ambiente orientado a serviços, os consumidores podem decidir mudar de provedor sempre que for necessário [ELFATATRY & LAYZELL, 2004]. Confiança constitui um elemento básico para se estabelecer parcerias, permitindo aos negociadores explorar novas possibilidades de contrato e correrem menores riscos, aumentando assim, a probabilidade de estabelecerem novos acordos.

2.5 Considerações Finais

Este capítulo foi dividido em quatro temas principais: (i) Os conceitos e propriedades intrínsecas do serviço; (ii) O paradigma de Computação Orientada a Serviços; (iii) O paradigma de uma Arquitetura Orientada a Serviços, e por fim, (iv) Negociação em Ambientes Orientados a Serviços. Para entender corretamente o conceito de Internet de Serviços, uma retrospectiva histórica aliada a uma identificação detalhada das especificidades dos serviços que podem ser digitalizados é fundamental. O segundo tema utiliza serviços como o elemento fundamental para o desenvolvimento e composição de aplicações distribuídas. O terceiro tema possibilita a comercialização de serviços em uma "Nova Internet". Serviços poderão ser facilmente criados, modificados e combinados para atender processos de negócio específicos. Finalmente, o quarto tema descreveu uma abordagem estruturada e baseada em um modelo orientado a serviços para projetar e lidar com a complexidade intrínseca dos serviços. Neste modelo, os elementos dos serviços são identificados, a prestação do serviço é negociada, o serviço é executado e então descartado.

Para que a Internet de Serviços se torne uma realidade, inúmeras áreas de pesquisa precisam ser reexploradas. Na área dos negócios, por exemplo, as contribuições sobre novos modelos de negócios e esquema de preços serão de grande importância. Já na área do direito, novas leis e assuntos relacionados com a oferta e

contratação de serviços suportados por dispositivos de informação e comunicação serão imprescindíveis.

Capítulo 3

Trabalhos Relacionados

Este capítulo aborda o estudo de três propostas de arquitetura que estão relacionadas à proposta desta dissertação. Duas propostas (CASCADAS e BIONETS) de redes futuras centradas em serviços e uma proposta (XIA) de núcleo misto, centrada em *hosts*, serviços e conteúdos.

3.1 CASCADAS

O *Component-ware for Autonomic Situation-aware Communications, and Dynamically Adaptable Services* (CASCADAS) é um projeto do *Sixth Framework Programme* (FP6) que teve início em Janeiro de 2006 e término em Dezembro de 2008. Foi coordenado por Antonio Manzalini e seus custos elegíveis chegaram a 6.920.795 Euros. O FP6 é um programa de pesquisa e desenvolvimento tecnológico da União Européia que tem como objetivo melhorar a integração e a coordenação da pesquisa na Europa financiando e promovendo a pesquisa e o desenvolvimento tecnológico.

De acordo com os idealizadores deste projeto, o principal objetivo do CASCADAS é desenvolver um ecossistema baseado em elementos autonômicos que permita a execução, composição e implantação de serviços inovadores, flexíveis e capazes de lidar com ambientes imprevisíveis. A proposta consiste em conceber uma "ecologia" de serviços no qual estes poderão prosperar e evoluir para satisfazer as necessidades específicas dos usuários.

CASCADAS prevê aplicações e serviços para interagir com usuários e ao mesmo tempo, avaliar as condições físicas e contextuais da rede atual. Para este fim, aplicações e serviços são desenvolvidos e implementados como *Autonomic Communication Elements* (ACEs) individuais ou agregados que dinamicamente se auto-organizam e interagem, a ponto de proporcionar a funcionalidade desejada.

3.1.1 ACE – Autonomic Communication Element

Os ACEs são abstrações de *software* para serviços de comunicação autonômica, que se organizam autonomamente com outros ACEs e se adaptam às mudanças do ambiente para atingir determinados objetivos. ACEs podem ser resumidos como sendo a base para a criação e evolução de serviços autonômicos em ambientes distribuídos [MANZALINI, et. al., 2009].

Sua arquitetura é inspirada no modelo biológico e envolve órgãos altamente interconectados, responsáveis por tarefas específicas, que juntos mantêm a entidade viva. Um ecossistema de ACEs possibilita fornecer serviços ou partes de um serviço a outros ACEs.

Cada ACE consiste de uma parte comum e uma parte específica. A parte comum contém seus órgãos (ver **figura 3.1**) e as funcionalidades comuns. A parte específica varia de ACE para ACE e compreende o Auto Modelo e as funcionalidades específicas de cada ACE. Desta forma, um desenvolvedor ao criar um ACE, só precisará de fato, criar a parte específica.

A **figura 3.1** redesenhada de [HASELOFF, et. al., 2008] ilustra a estrutura do ACE a partir de um conjunto de órgãos com funções bem definidas.

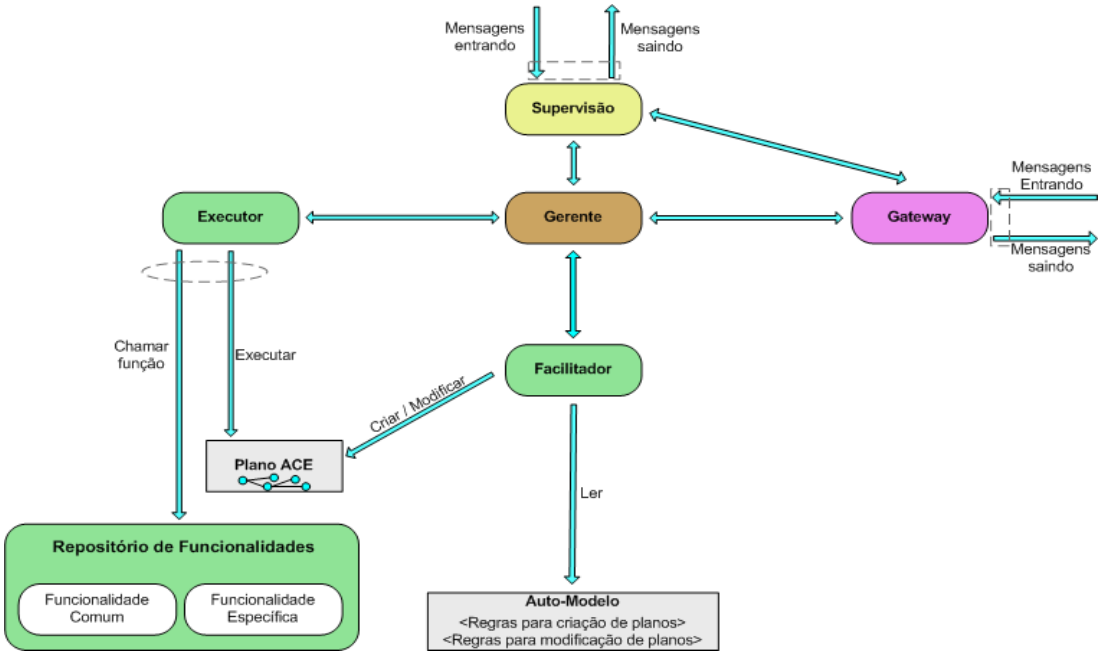

Figura 3.1: *Arquitetura do ACE [HASELOFF, et. al., 2008].*

Conforme ilustrado na **figura 3.1**, um ACE é composto por seis órgãos (Gateway, Gerente, Facilitador, Executor, Repositório de Funcionalidades, e por fim, Supervisão), cada um deles desempenhando uma determinada tarefa. Destes, três são essenciais para que o serviço seja disponibilizado: (*i*) o Facilitador, responsável por criar, analisar e modificar os planos, sendo este o principal órgão de adaptabilidade; o (*ii*) Executor, responsável por executar os planos criados e o (*iii*) Repositório de Funcionalidades, responsável por armazenar funcionalidades comuns e específicas. A seguir, é descrita a funcionalidade de cada órgão.

Gateway: É responsável pela comunicação e possíveis interações entre ACEs. Para que seja possível desempenhar esta tarefa, dois protocolos de comunicação são utilizados: (*i*) um protocolo sem conexão (*GN - Goal Needed* / *GA – Goal Achievable*) [HOFIG, et. al., 2006] é usado para descobrir os serviços, através do paradigma publica/assina (que será visto em maiores detalhes no capítulo 4 - seção 4.2) e (*ii*) um protocolo de comunicação orientado a conexão baseado no *framework* DIET (*Decentralised Information Ecosystem Technologies*) [MARROW, et. al., 2001], usado para estabelecer canais de comunicação dedicados entre ACEs através de sua contratação.

Gerente: Gerencia a comunicação interna e o ciclo de vida do ACE. O modelo de comunicação interna utiliza o paradigma publica/assina por meio de eventos. Todos os órgãos do ACE podem publicar eventos para o "barramento de eventos" (que é uma parte do gerente) e podem se inscrever para receber eventos de um determinado tipo. Caso um evento de um determinado tipo seja publicado, o gerente irá entregá-lo a todos os órgãos que assinaram esse tipo de evento. Além da comunicação interna, este órgão gerencia o ciclo de vida. A gerência do ciclo de vida compreende etapas como programação, controle e execução de operações. Caso seja necessário realizar alguma operação no ciclo de vida, o gerente comunicará todos os órgãos sobre a ação a ser tomada e só a executará depois que todos confirmarem a sua disponibilidade [MANZALINI, et. al., 2009].

Facilitador: Cria planos para adaptar o comportamento do ACE às mudanças detectadas. Um Plano é uma sequência de ações que devem ser realizadas para alcançar um objetivo específico, podendo um ou mais planos resultar em um serviço. Cada ACE possui um plano inicial que foi definido pelo desenvolvedor dentro do Auto Modelo e criado pelo Facilitador quando o ACE foi inicializado. Quando o ACE é inicializado, o Facilitador carrega o Auto Modelo e o analisa durante toda a vida [MANZALINI, et. al., 2009]. Com base nas análises feitas, o Facilitador pode modificar ou finalizar planos, bem como criar novos. A **figura 3.2** redesenhada de [BENKŐ, et. al., 2008] ilustra o processo realizado.

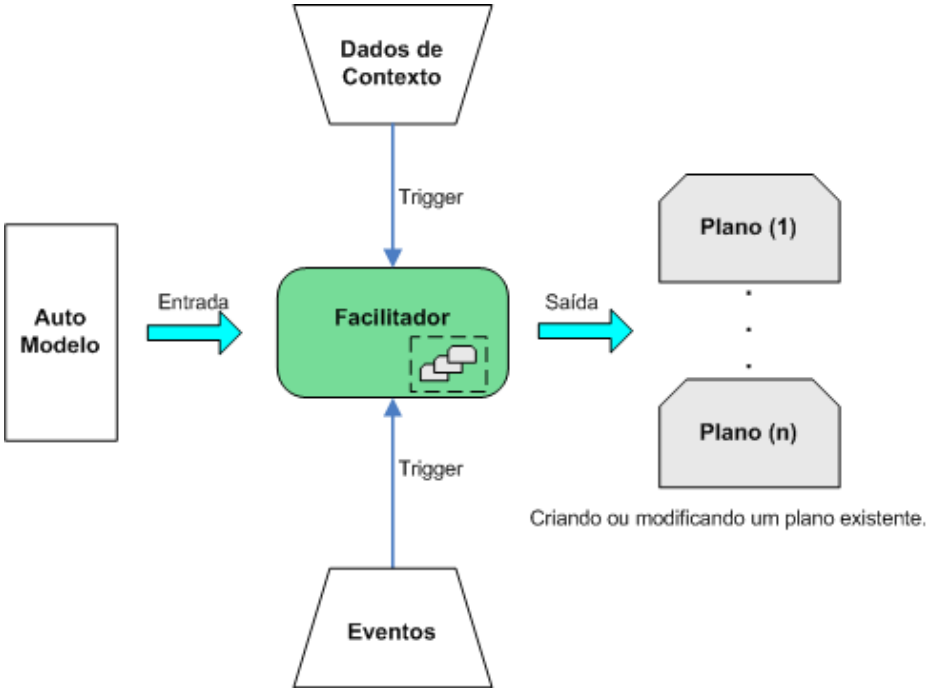

Figura 3.2: *Modelo de processo do órgão Facilitador [BENKŐ, et. al., 2008].*

Conforme ilustrado na **figura 3.2**, o Facilitador cria planos com base em três variáveis: (*i*) Auto Modelo, que é um conjunto de regras e procedimentos XML fornecido pelo desenvolvedor durante a criação do ACE, que irá descrever seu comportamento, incluindo regras para a sua modificação e adaptação. Desta forma, cada ACE possui seu próprio modelo que é carregado na fase de inicialização e é continuamente analisado pelo Facilitador durante a vida do ACE; (*ii*) Dados de Contexto, que é qualquer informação de contexto que possa ser obtida pelo ACE em atividade, fazendo com que o Facilitador crie novos planos, modifique ou finalize planos existentes contidos no Auto Modelo; e por fim, (*iii*) os Eventos, procedimentos de operação predefinidos, que quando disparados, possibilitam ao Facilitador criar um plano para modificar o comportamento do ACE [HASELOFF, et. al., 2008]. Resumindo, o Facilitador pode ser usado para criar um novo plano, parar a execução ou alterar um plano existente, bem como apresentar um novo Auto Modelo para substituir o atual.

Executor: Executa os planos que foram criados pelo Facilitador. O Executor pode executar vários planos em paralelo, possibilitando um ACE atender solicitações de vários ACEs. Com tantos planos, é imprescindível que o Executor consiga identificar os planos que foram executados, sendo necessário que cada plano

possua seu próprio ID, podendo existir apenas uma cópia com mesmo ID. Quando um novo plano com o mesmo ID é apresentado ao Executor, a cópia antiga é substituída por uma nova. Esta situação pode acontecer quando um plano que já está ativo é modificado pelo Facilitador.

Repositório de Funcionalidades: Armazena as funcionalidades que podem ser utilizadas pelo ACE para fornecer um serviço. É constituído por funcionalidades comuns (disponíveis em cada ACE) e específicas. A Funcionalidade Comum é uma característica importante para auto-similaridade e compreende todas as funcionalidades necessárias para manter o ACE "vivo", como por exemplo, estabelecer contratos para fornecimento de serviços. Já a Funcionalidade Específica, armazena funcionalidades ou serviços específicos que um ACE pode fornecer a outros, como por exemplo, os serviços criados de forma autonômica ou manual serão disponibilizados por meio deste repositório.

Supervisão: monitora e registra as operações do ACE com base no Auto Modelo e outros dados que o sistema de supervisão julgue importante. Caso um ACE esteja em uma situação indesejável, o órgão de supervisão poderá definir medidas corretivas para que ele volte ao estado normal. Para que isso seja possível o órgão de supervisão conta com a cooperação dos órgãos de gerência e *gateway*.

3.1.2 Rede de Conhecimento

A rede de conhecimento (*KN - Knowledge Network*) é uma estrutura genérica que organiza o conhecimento distribuído independente do seu formato em um sistema que vai permitir que ele seja recuperado de forma eficiente. A proposta é que a KN atue como uma camada intermediária que se conecta a uma variedade de fontes, as organiza com base em diferentes conceitos, oferecendo o conhecimento de forma bem estruturada para serviços e aplicações individuais [MANZALINI, et. al., 2009]. Isto é, esta camada é responsável por receber dados, analisá-los, processá-los e construir um contexto compacto e de alto nível, evitando que serviços pervasivos tenham que acessar e processar grandes quantidades de dados. A **figura 3.3**

redesenhada de [BAUMGARTEN, et. al., 2008] ilustra a arquitetura da KN e seus três níveis.

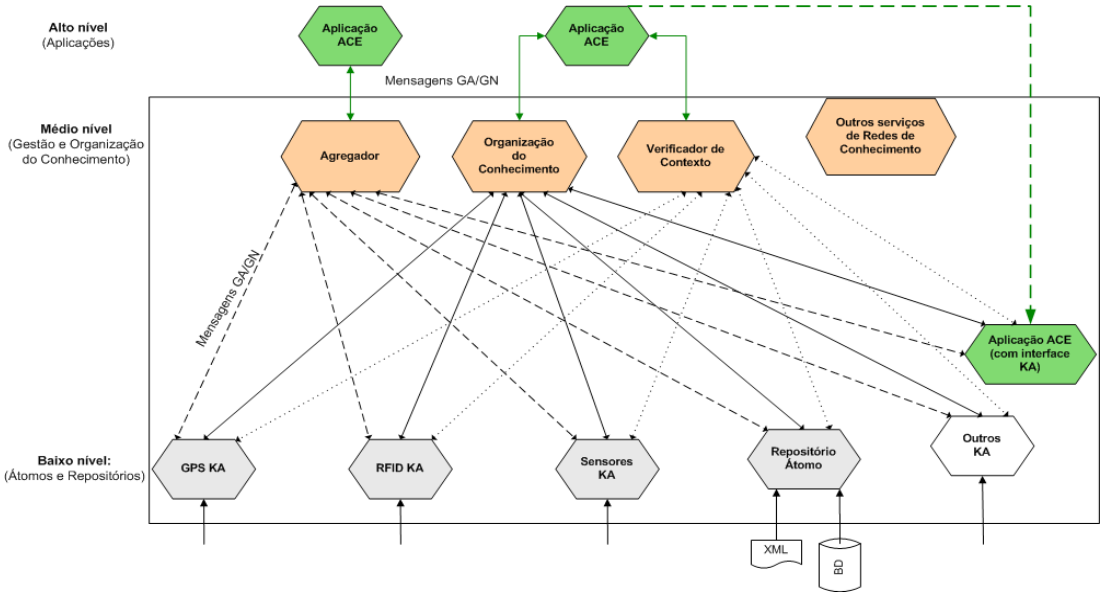

Figura 3.3: *Arquitetura da Rede de Conhecimento [BAUMGARTEN, et. al., 2008].*

O nível baixo é composto por dois componentes, *Knowledge Atom* (KA) que são os dados que formam o conhecimento, e *Knowledge Container* (KC) que são os repositórios responsáveis por organizar esses dados. Ambos são implementados como ACEs. Para que os ACEs do nível alto possam se tornar parte deste nível, é necessário que estes tenham uma interface KA [BAUMGARTEN, et. al., 2008].

O nível médio trata o conhecimento e não mais dados. Neste nível os elementos de dados gerados pelo nível baixo são analisados para construir um contexto compacto e de alto nível. Depois que os dados são introduzidos na KN, através do conceito de KAs, eles serão organizados com base em diferentes componentes, tais como, agregação, organização do conhecimento e verificação do contexto.

O componente agregador possibilita estabelecer relações entre KAs e armazenar dados quando houver uma solicitação. A Organização do Conhecimento é responsável por organizar os dados em KCs e criar uma interface para realizar consultas baseadas em conceito (ou seja, palavras chave), permitindo que ACEs do nível alto acessem informações baseadas em conceito. E, por fim, o Verificador de Contexto é responsável por verificar a consistência das informações na KN de acordo

com parâmetros configuráveis. Isto é feito acessando os KAs e consultando componentes de organização do conhecimento a fim de verificar a consistência do conhecimento. De acordo com o resultado desta verificação, é possível notificar o aplicativo sobre problemas [BAUMGARTEN, et. al., 2008]. Os componentes citados são alocados e implementados dentro de uma KN, oferecendo diferentes maneiras para organizar, consultar e gerir o conhecimento.

Finalmente, no nível alto, aplicativos e serviços baseados em ACE podem descobrir a presença de componentes de KN pela simples emissão de uma mensagem GN, podendo consultar os componentes mais adequados para obter informações sobre situações ao redor [BAUMGARTEN, et. al., 2008]. Este nível faz a ponte entre os serviços individuais e aplicações que podem utilizar o conhecimento fornecido pela rede.

Recapitulando, assim que inseridos na KN, os KAs são automaticamente processados, combinados uns com os outros através do nível médio e disponibilizados a serviços e aplicações no nível alto por meio de uma interface dedicada a consulta.

Considerando a grande diversidade de fontes de dados, como sensores e dispositivos, até mesmo redes pequenas podem gerar grandes quantidades de dados, não sendo interessante criar uma única KN para tratar, analisar e gerenciar dados de diferentes ambientes. Diferentes tipos de serviços podem ter diferentes necessidades em termos de qual tipo de conhecimento é necessário. Desta forma, o projeto CASCADAS considera criar uma variedade de KNs, que irão coexistir dentro em um espaço de conhecimento global, sendo cada rede limitada às fronteiras do seu conhecimento.

3.1.3 CNQ Service (Cross Network Query Service)

O *Cross Network Query Service* é um mecanismo de recuperação do conhecimento baseado em um mecanismo de busca que utiliza as descrições semânticas fornecidas através da interface KA [BAUMGARTEN, et. al., 2008].

Enquanto as KNs organizam os KAs de acordo com suas descrições individuais, o serviço de consulta utiliza o mesmo conjunto de informações para identificar e extrair o conhecimento desejado a partir da rede. Desta forma, a KN é usada como uma camada intermediária entre usuário e fornecedores de conhecimento que não só possibilitam o acesso ao conhecimento, mas também extraem novos conhecimentos, que poderão ser novamente publicados no âmbito da KN.

O *CNQ Service* é transparente para o usuário, que se desejar, pode utilizá-lo para acessar diretamente as fontes de conhecimento. A **figura 3.4** redesenhada de [BAUMGARTEN, et. al., 2008] ilustra o esquema deste mecanismo.

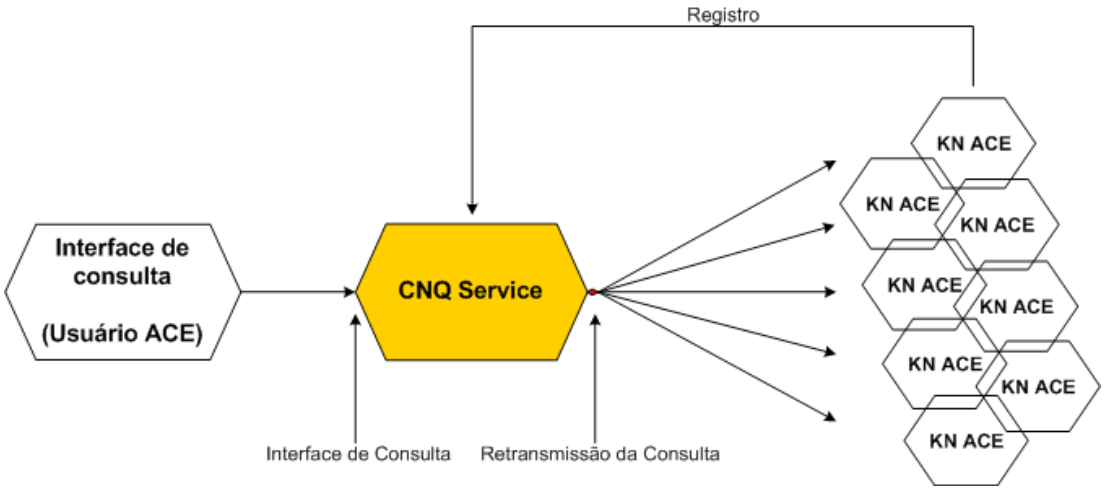

Figura 3.4: *Cross Network Querying Service [BAUMGARTEN, et. al., 2008].*

Conforme ilustrado na **figura 3.4**, um usuário ou aplicação ACE pode consultar o *CNQ Service*, que é implementado como um ACE diferenciado que possui uma interface de consulta idêntica à interface de consulta das KNs. Desta forma, o processo de consulta realizado diretamente às KN individuais ou através do *CNQ Service* é idêntico.

Dado que uma consulta foi realizada por meio do *CNQ Service*, esta será retransmitida a todas as redes de conhecimento registradas. Após todas as respostas serem recuperadas, o *CNQ Service* irá fundi-las em um único conjunto de resultados que será disponibilizado para o usuário que iniciou a consulta. Se existirem duplicatas, estas serão removidas do resultado. Para melhorar a resiliência do sistema, um mecanismo de tempo limite (*time out*) foi implementado para garantir que o serviço seja capaz de responder ao usuário ou aplicação ACE, mesmo que nem

todas as KNs tenham respondido ao pedido da consulta. Assim, o serviço pode ser garantido mesmo que uma KN se torne temporariamente indisponível [BAUMGARTEN, et. al., 2008].

3.1.4 Descoberta e Contratação

A descoberta e contratação de ACEs são essenciais para que seja possível criar planos e serviços que utilizem um conjunto de ACEs. A descoberta é o processo de identificação e seleção de um ACE apropriado. A contratação por sua vez, é o estabelecimento de uma conexão entre o ACE requerente e o selecionado. A **figura 3.5** redesenhada de [BENKŐ, et. al., 2008] ilustra um fluxo de mensagens típicas para estas duas etapas.

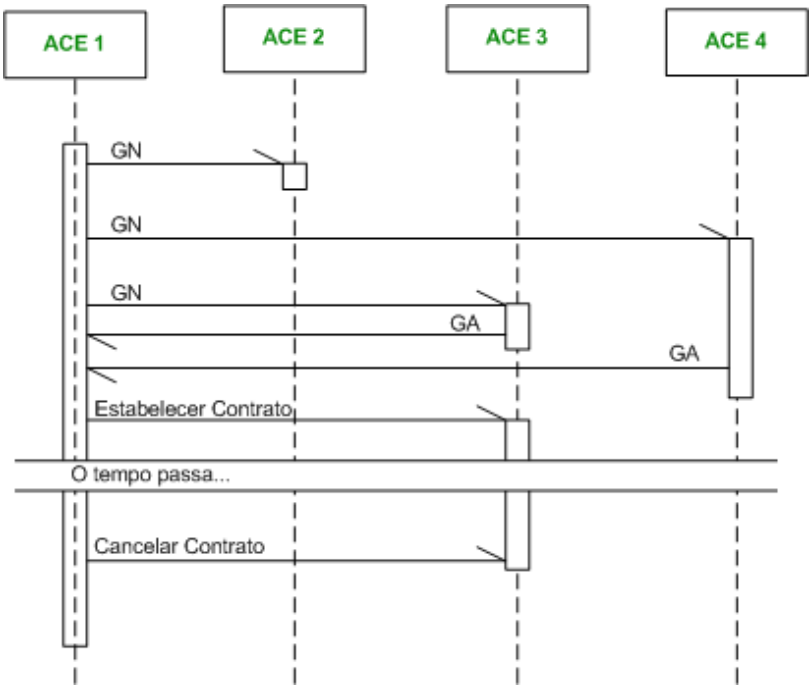

Figura 3.5: *Descoberta e Contratação [BENKŐ, et. al., 2008].*

Conforme ilustrado na **figura 3.4**, a primeira etapa é realizada quando o ACE 1 envia uma mensagem *Goal Needed* (GN) a todos os ACEs habitantes do domínio semântico (ambiente composto por ACEs com funcionalidades específicas similares) que lhe interessa. Esta mensagem é uma espécie de pedido, com uma descrição semântica em anexo, que especifica que tipo de funcionalidades o ACE solicitane necessita de outros ACEs para atingir seus objetivos [MANZALINI,

2007]. Os demais ACEs ao receberem o pedido, poderão responder utilizando uma mensagem *Goal Achievable* (GA), utilizada para informar semanticamente a outros ACEs seu endereço e qual tipo de tarefa um determinado ACE é capaz de fornecer. Após a identificação dos ACEs aptos a atender a solicitação, o ACE 1 seleciona aquele que julga mais adequado.

Observe que até o momento não existe nenhuma conexão estabelecida entre eles, foi realizada apenas a descoberta de ACEs potenciais, bem como sua seleção. Para que seja possível estabelecer a conexão e posteriormente o provimento do serviço, faz-se necessário que seja estabelecido um contrato entre as entidades envolvidas. O ACE 1 então, envia ao ACE selecionado (ACE 3) o evento "Estabelecer contrato", que foi previamente definido na funcionalidade comum de cada ACE da arquitetura. Assim que o ACE 3 aceitar o contrato, este será identificado e armazenado no repositório de funcionalidades dos ACEs envolvidos, mais especificamente na Sessão Global, que é uma sessão inicialmente criada como funcionalidade comum, mas que atende funcionalidades específicas, como por exemplo, o armazenamento de contratos.

A partir deste momento, é estabelecida uma conexão entre as entidades envolvidas, o serviço poderá ser criado, identificado e relacionado ao contrato estabelecido. É importante destacar que qualquer ACE é capaz de solicitar o estabelecimento de contratos, desde que tenha recebido eventos GA de outros ACEs.

Caso o contrato seja cancelado por uma das partes envolvidas, a conexão entre os ACEs é desfeita e nenhuma comunicação direta será mais possível, sendo que todas as tentativas de utilizá-la resultarão em um erro.

3.2 BIONETS

O *BIOlogically-inspired NETworks and Services* (BIONETS) é um projeto do *Sixth Framework Programme* (FP6) que teve início em Janeiro de 2006 e término em Dezembro de 2009. Foi coordenado por Daniele Miorandi e seus custos elegíveis chegaram a 6.948.333 Euros.

De acordo com os idealizadores deste projeto, a motivação para BIONETS veio das tendências da computação pervasiva e ambientes de comunicação caracterizados por vários dispositivos conectados em rede, como por exemplo, roupas, carros, eletrodomésticos, eletrônicos, entre outros. Essa grande diversidade de dispositivos contribui para o crescimento e complexidade da nova rede, superando a Internet atual e fazendo com que seja necessário se ter novas abordagens que tratem não só a heterogeneidade de recursos e nós, mas também os futuros problemas de escalabilidade e complexidade [BIONETS, 2011].

O objetivo do projeto é desenvolver uma rede completamente integrada, onde os serviços sejam capazes de se adaptar a um grande número de dispositivos heterogêneos, a mudanças de ambientes e evoluir de forma autônoma, sendo a arquitetura inteiramente distribuída e descentralizada, resistente a falhas de rede, ataques e desconexões, e, que os serviços sejam capazes de evoluir sem qualquer tipo de intervenção humana.

3.2.1 Arquitetura de Serviços

Em BIONETS, os serviços são definidos como entidades que podem fornecer o conhecimento, conteúdo ou funcionalidade para outros serviços e usuários. Dentro deste contexto, as aplicações de usuários, serviços de aplicativos, e serviços de protocolo podem ser definidos como composições de outros serviços ou células de serviços **[LINNER, et. al., 2007]**.

BIONETS descreve uma arquitetura para sistemas de computação autonômica[1] que pode ser dividida em três partes: Estrutura de Serviços, que inclui os serviços, e as funções de apoio à sua execução, distribuição, pervasividade e gestão. Estrutura de Interação, cuja finalidade é disponibilizar modelos de interação simultânea, apoiando a comunicação entre os diversos serviços distribuídos. Estrutura de Rede, que fornece os recursos básicos de comunicação. A **figura 3.6** redesenhada de **[LINNER, et. al., 2007]** ilustra esta divisão.

[1] **Computação Autonômica** é um conceito que reúne diversas áreas da computação com o objetivo de criar sistemas de computação com características de autogestão [HUEBSCHER and McCANN, 2008].

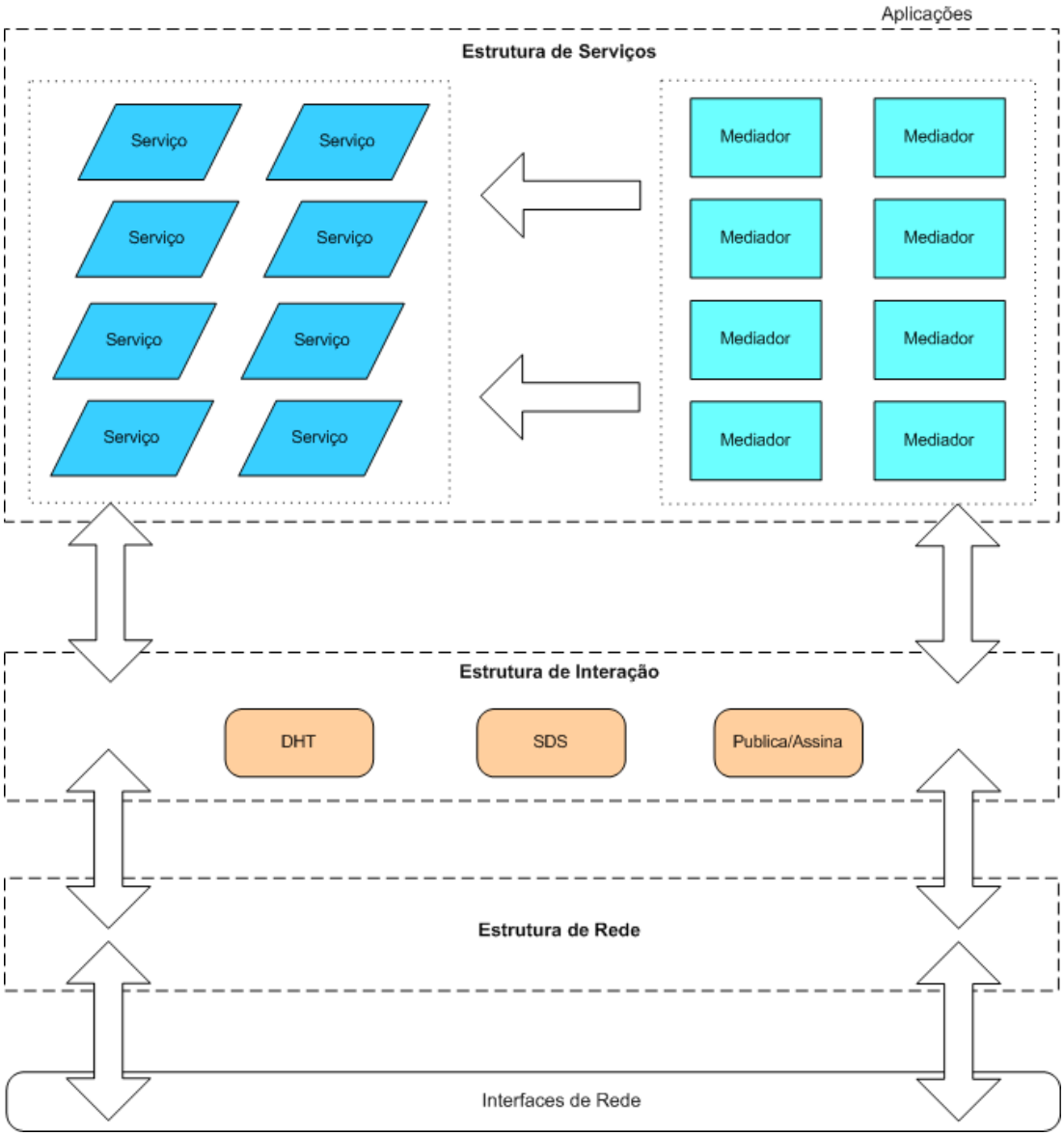

Figura 3.6: *Estruturas de Serviços, Rede e Interação [LINNER, et. al., 2007].*

Estrutura de Serviços: Possibilita a execução dos serviços nos nós, e, por meio de estratégias de adaptação e evolução, possibilita que os serviços reajam de forma autonômica às mudanças no ambiente. Este tipo de autonomicidade pode ser tratada pelo nó e pelo serviço. No nó possui melhor desempenho em relação à complexidade e escalabilidade, embora a autonomicidade do serviço tenha uma maior flexibilidade. Para possibilitar uma maior flexibiliade pelo nó, foram criados os mediadores de serviços, entidades capazes de interagir com os serviços por meio dos modelos de interação disponibilizados pela Estrutura de Interação, mesmo que eles estejam hospedados em nós diferentes [LINNER, et. al., 2007].

Estrutura de Interação: Disponibiliza modelos de interação que possibilitam desacoplar a Estrutura de Serviços dos protocolos de comunicação subjacentes, nomeando e endereçando esquemas e características da rede. Os principais modelos citados [LINNER, et. al., 2007] são: Espaço de Dados Semânticos (SDS – Semantic Data Space), Tabelas Hash Distribuídas (DHT - Distributed Hash Tables), e o paradigma Publica/Assina (Publish/Subscribe).

Estrutura de Rede: é responsável por fornecer os meios de comunicação adequados para promover a evolução do serviço, mesmo na presença de redes de larga escala, heterogêneas e particionadas, devendo as interfaces de rede estar aptas a lidar com o contexto de redes desaparecendo, redes que podem estar disponíveis em um momento e indisponíveis no momento seguinte [LINNER, et. al., 2007].

3.2.2 Arquitetura de Rede

BIONETS é construída em torno de uma arquitetura de rede de dois planos (nós U e nós T). Os dispositivos de nível mais baixo, conhecidos como nós T (*Tiny-Nodes*), são usados para realizar a interface com o ambiente e coletar informações contextuais, enquanto os dispositivos de nível superior, conhecidos como nós U (*User-Nodes*), possibilitam o usuário interagir com o sistema. Em termos de função na arquitetura, nós T atuam como fontes de dados, enquanto os nós U atuam como consumidores de dados. A **figura 3.7** redesenhada de [TAHKOKORPI, et. al., 2006] ilustra as camadas desta arquitetura.

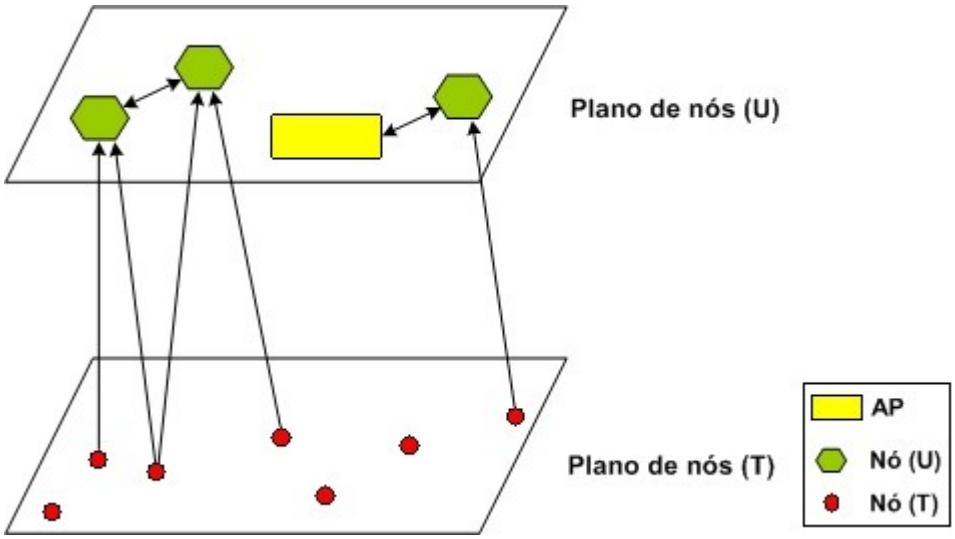

Figura 3.7: *Arquitetura de duas camadas [TAHKOKORPI, et. al., 2006].*

Observe na **figura 3.7** que além dos elementos citados, esta arquitetura pode conter um terceiro elemento, o AP (*Access Point*). Para melhor entendimento, todos os elementos serão descritos a seguir.

Nó T: é um dispositivo simples, barato, com poder de detecção e identificação, mas, pouquíssimo poder de processamento, armazenamento e comunicação. Realiza a *interface* com o ambiente e é utilizado para fornecer a ciência de contexto aos serviços e nós de usuários. Os nós T não se comunicam entre si, mas apenas com nós U nas proximidades. Esta é a diferença básica e importantíssima que descaracteriza a abordagem convencional de redes *ad hoc*, onde, todos os nós executam operações de armazenamento e transmissão [TAHKOKORPI, et. al., 2006].

Nó U: é um dispositivo complexo e poderoso, mas incapaz de interagir diretamente com o ambiente, utilizando os nós T para atingir este objetivo. Diferentemente dos nós T, os nós U podem se comunicar entre si e com os nós T. A comunicação entre os nós U é oportunista, devendo estar na mesma faixa de comunicação e que um dos dispositivos requeira a interação. Alguns exemplos de nós de usuários são: *Laptops* e *smartphones* [TAHKOKORPI, et. al., 2006].

AP: é um dispositivo complexo e poderoso que atua como *gateway* com o mundo IP. APs não executam serviços, apenas permitem a interoperabilidade entre ambientes IP e BIONETS.

3.2.3 Evolução dos Serviços

De acordo com o paradigma BIONETS a evolução dos serviços é baseada em organismos digitais vivos, onde o nascimento é a criação do serviço, a reprodução é a evolução do serviço e a morte é a sua desativação.

No ramo da biologia, evolução refere-se à mudança das características hereditárias de uma espécie de uma geração para outra. Este processo faz com que surjam novas espécies mais adaptadas ao longo do tempo. Em termos de serviços, a evolução do serviço implica em uma adaptação de longo prazo para as mudanças no ambiente, podendo o serviço adquirir novas funcionalidades [TSCHUDIN, et. al., 2005] e [MIORANDI, et. al., 2006]. Foram investigadas as seguintes ordens de evolução de serviços:

Evolução de Ordem "0": Quando a evolução acontece sobre um conjunto predefinido de parâmetros que determinam o comportamento do sistema, permitindo que os serviços evoluam para se adaptar às mudanças do ambiente, sendo esta evolução inspirada por Algoritmos Genéticos[2] (*Genetic Algorithm*). Em cada nó, um genótipo descreve o esquema de encaminhamento usado, um processo de seleção favorece a difusão dos genótipos[3] mais aptos, sendo novos genótipos criados através da combinação dos já existentes. Todo sistema é projetado de tal forma a apresentar uma tendência a níveis mais elevados de aptidão [LINNER, et. al., 2007].

No ramo de Algoritmos Genéticos, os indivíduos são avaliados por uma função de avaliação que irá dar uma "nota" de acordo com seu nível de aptidão

[2] **Agorítmos Genéticos** é uma parte dos algoritmos evolucionários e se baseia em conceitos da genética e da evolução de populações de seres vivos. O algoritmo trata as possíveis soluções de um problema e as classifica como indivíduos de uma população, que irá evoluir a cada geração, se comportando de forma parecida com a teoria da evolução das espécies definida por Charles Darwin.
[3] **Genótipo** na área de Algoritmos é a estrutura ou representação interna usada para armazenar os parâmetros a serem otimizados.

(*fitness*). Isto é, quanto mais próximo da solução ótima o individuo estiver, melhor é a avaliação e consequentemente, melhor a "nota", onde os indivíduos mais aptos irão participar do processo reprodutório, podendo este ser realizado de duas formas: recombinação (*crossover*) e mutação.

Evolução de Ordem "1": É quando a evolução ocorre por meio da composição e adaptação de novos serviços. A idéia se baseia em componentes de serviço de fraco acoplamento, que podem ser combinados e orquestrados em tempo de execução e conforme a demanda, a fim de proporcionar novas funcionalidades com um excelente desempenho. Este processo se baseia em uma estrutura de árvore que descreve o modelo de composição atual, sendo a árvore resultante considerada como o genótipo que descreve o serviço atual. Provavelmente, este processo evolutivo será baseado em técnicas da área de Programação Genética[4] (Genetic Programming) [LINNER, et. al., 2007].

Evolução de Ordem "2": É a forma de evolução mais desafiadora prevista no ambiente BIONETS. O objetivo é proporcionar a autogeração de serviços e protocolos de rede. A noção de "*software* autocatalístico" consiste de que os programas são modelados como moléculas que regulam a sua própria produção e consumo. Este modelo inclui um repositório de códigos responsável por armazenar os "genes" [5], que poderão ser injetados em um abiente adequado para se tornarem programas a serem executados [LINNER, et. al., 2007].

Independente da ordem de evolução, do ponto de vista de rede é importante ressaltar que os nós T não participam da interação evolutiva, mas podem influenciar a evolução através do sensoriamento do ambiente e fornecimento de dados a nós de usuários, e, naturalmente ao sistema nele contido. Já os nós U são indispensáveis para que a evolução ocorra, pois os serviços podem migrar de um dispositivo para outro e sofrer mutações, dando origem a novos serviços.

---

[4] **Programação Genética** é um algoritmo evolutivo inspirado biologicamente que resolve os problemas automaticamente sem a intervenção de usuários. É uma especialização dos algoritmos genéticos onde cada indivíduo é um programa de computador. A idéia deixa de ser evoluir indivíduos ou cromossomos, e passa a ser evoluir programas.

[5] **Gene** é a unidade fundamental da hereditariedade.

3.2.4 Interoperabilidade com Redes IP Legadas

Embora BIONETS não dependam de nenhuma infraestrutura para funcionarem, elas podem explorar oportunisticamente a presença de redes IP para obterem uma melhor qualidade e variedade de serviços. Da mesma forma, serviços legados podem aproveitar BIONETS para coletarem dados ambientais [LINNER, et. al., 2007].

Para que seja possível estabelecer a comunicação entre os ambientes BIONETS e IP é utilizado um AP, que conta com a presença de um servidor proxy capaz de traduzir as operações BIONETS para as redes IP e vice-versa. Segundo [PELLEGRINI, et. al., 2007] esta interoperabilidade é limitada a três tipos de operações:

Recuperação de dados remotos localizados na rede IP: Um nó de usuário ao se registrar junto a um AP receberá um endereço que será vinculado ao seu nome. A partir deste momento ele será capaz de se comunicar com ambientes IP, recuperando ou enviando dados para um *host* remoto. Por se tratar de uma conexão *wireless*, caso o nó de usuário saia da área de comunicação do AP, a conexão será perdida.

Recuperação de dados do ambiente através da rede IP: Dispositivos baseados em IP poderão acessar dados de ambientes específicos através do envio de uma consulta para o AP, que, utilizando o servidor *proxy*, irá traduzi-la para o formato BIONETS e enviá-la para nós de usuário passando.

Tunelamento de consultas: A infraestrutura IP pode ser usada para conectar diferentes locais BIONETS. Esta operação pode ser uma opção interessante quando a mobilidade dos nós não é suficiente para oferecer uma rápida difusão dos dados. Entretanto, para viabilizar esta operação, os APs deverão utilizar algum serviço que possibilite realizar consultas e construir uma rede sobreposta, utilizada para encaminhar mensagens entre os nós BIONETS de localidades distintas [LINNER, et. al., 2007].

3.3 XIA

A *eXpressive Internet Architecture* (XIA) é um projeto que aborda a crescente diversidade de modelos de uso da rede. Foi iniciado pela *Carnegie Mellon University* e recentemente selecionado pela junta de *Computer and Information Science and Engineering* (CISE) da *National Science Foundation* (NSF) e inserido como parte do programa *Future Internet Architecture* (FIA).

O XIA é uma proposta de rede futura centrada no essencial (*principal-centric*). O centro da arquitetura é assim definido devido agrupar as abordagens essenciais pesquisadas para redes futuras, podendo uma entidade ser nomeada como um *host,* um domínio, um serviço, ou parte de um conteúdo específico.

De acordo com os idealizadores do projeto, os objetivos do XIA são preservar os pontos fortes da arquitetura centrada em *hosts*, melhorar a segurança e construir uma rede que seja capaz de suportar nativamente os modelos de redes centradas em conteúdo, serviço e *host*, e, que possa evoluir para suportar outros modelos de comunicação, como por exemplo, usuários e grupos.

Uma premissa desta arquitetura é que todos os objetos nela contidos sejam intrinsecamente seguros. A identificação destes objetos deverá ser realizada por meio de identificadores expressivos (XIDs) de 160 *bits* que serão utilizados como origem e destino para o encaminhamento de pacotes na rede. Para diferenciar significativamente as identificações dos objetos, são utilizados os termos HID, CID, e SID para se referir aos identificadores expressivos de *hosts*, conteúdos e serviços, respectivamente. A nomeação do conteúdo é baseada no *hash* do conteúdo; a nomeação do serviço e *host* é baseada no *hash* de suas respectivas chaves públicas, utilizando assim, identificadores seguros que evitam, por exemplo, a falsificação de endereços na rede [ANAND, A., et. al., 2011].

Se comparado com os demais trabalhos relacionados, a distinção mais visível, é que o núcleo do XIA aborda diferentes modelos e é totalmente flexível, não sendo fixa a utilização de uma base específica (serviços, por exemplo). XIA permite que soluções da arquitetura atual (centrada em *hosts*) sejam aliadas a novas propostas

de arquitetura (centrada em serviços, conteúdo e usuário, por exemplo), permitindo uma maior flexibilidade de suas entidades e ativos de rede, como por exemplo, roteadores.

3.3.1 Núcleo da Arquitetura

A rede XIA que também suporta *hosts*, até o momento é formada por dois outros modelos considerados essenciais: serviços e conteúdo.

Independentemente dos modelos que estejam no núcleo da arquitetura, estes deverão conter quatro características primordiais: (i) a semântica para que as entidades possam expressar sua intenção de comunicação com um objeto de modelo específico, como por exemplo, uma entidade que deseje pesquisar um conteúdo específico ou utilizar um determinado serviço; (ii) um XID único e (iii) um método para mapeá-lo, características utilizadas para gerar identificadores e nomes de objetos utilizados em rede. Estes deverão ser gerados de forma distribuída, ser únicos e planos, mesmo que, possam ser alcançados por meios hierárquicos; por fim, (iv) o processamento e encaminhamento de pacotes de qualquer tipo específico deve ser coordenado e distribuído, podendo otimizações de rede ser tratadas localmente em cada roteador (alternando sua base de comunicação, por exemplo) [ANAND, et. al., 2011].

3.3.2 Arquitetura

O projeto XIA é estruturado com base em três pilares [ANAND, et. al., 2011]: modelos essenciais, *fallbacks* e identificadores intrinsecamente seguros, que são descritos abaixo.

(*i*) **modelos essenciais** (*hosts,* conteúdos e serviços) devem possibilitar aos usuários e aplicações expressarem sua intenção, dando à rede uma flexibilidade significativa. Devem ser capazes de evoluir, ou seja, deverá ser possível acrescentar e/ou modificar modelos que compõem o núcleo da arquitetura com uma complexidade razoavelmente baixa. A adição de novos modelos deverá ser realizada

de forma incremental, como por exemplo, fornecendo suporte parcial às redes, e, posteriormente, suporte global;

(*ii*) a arquitetura deve introduzir novas funcionalidades de forma transparente e gradual, evitando problemas de utilização. Um grande desafio do XIA é como um roteador legado deverá lidar com a adição de um novo modelo considerado essencial, mas que não é reconhecido por ele. Para resolver esta questão, foi introduzido o conceito de retornos ou *fallback*. ***Fallbacks*** permitem as entidades especificarem ações alternativas (por exemplo, baseada em *hosts*), caso os roteadores não operem sobre a intenção primária (por exemplo, baseada em conteúdo). Por exemplo, um conteúdo pode ser recuperado através de sua identificação ou por meio de um identificador de *host* conhecido. Isto é, o roteador inicialmente irá operar sob a intenção primária (identificador do conteúdo), mas, caso não suporte este modelo, a funcionalidade *fallback* possibilitará que este utilize uma intenção secundária (identificador de *hosts*) para recuperar o conteúdo desejado a partir de um servidor conhecido. A possibilidade de interação entre os diversos modelos evita a sobrecarga de indireções, permite à rede evoluir e ganhar expressividade, não sendo limitada ao modelo legado (arquitetura centrada em *hosts*) [ANAND, et. al., 2011].

(*iii*) os **identificadores devem ser intrinsecamente seguros**, permitindo às entidades validarem se estão se comunicando com o modelo essencial correto, bem como diferenciar suas informações semânticas. Por exemplo, o requisito fundamental na comunicação baseada em *host* é autenticar os respectivos *hosts*, enquanto que na recuperação de conteúdo, é garantir a integridade e validade dos dados obtidos. Desta forma, identificadores XIA intrinsecamente seguros para *host* e conteúdo deverão ser obtidos de alguma forma, sendo capazes de refletir estes requisitos respectivamente.

Como apoio a estes requisitos, o XIA utiliza dois componentes que, juntos, possibilitam a comunicação fim a fim. O primeiro é o *eXpressive Internet Protocol* (XIP), protocolo de camada de rede utilizado por todos os modelos essenciais e proposto para substituir o atual *Internet Protocol* (IP), e, o segundo é o roteamento por salto, ambos não detalhados neste trabalho por não fazerem parte do escopo da proposta.

A **figura 3.8** redesenhada de [*EXPRESSIVE INTERNET ARCHITECTURE*, 2011] ilustra o núcleo da Arquitetura de Internet eXpressiva.

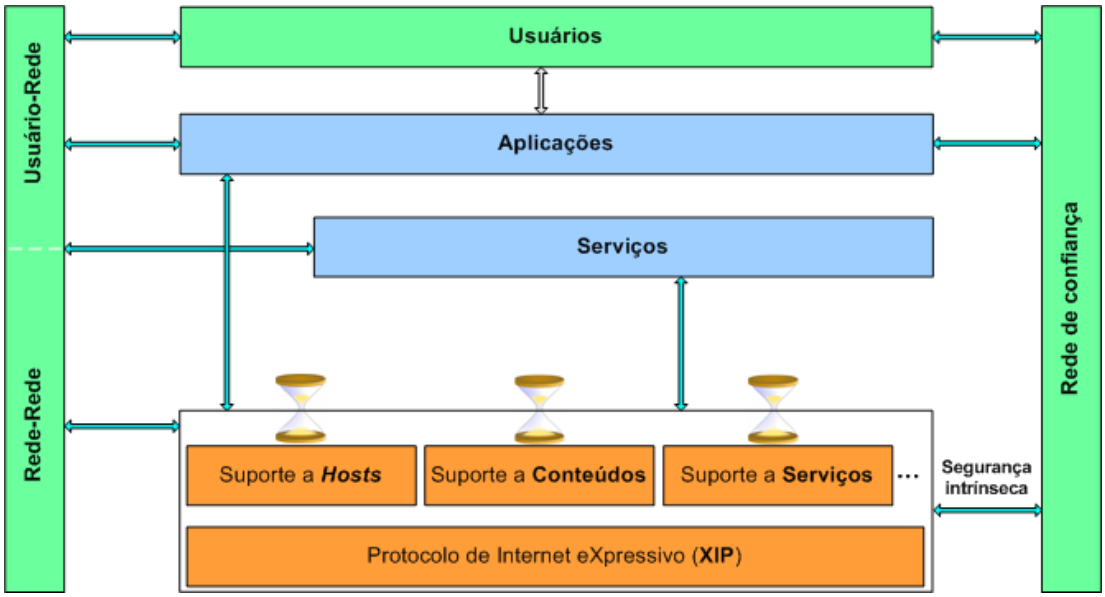

Figura 3.8: *Arquitetura XIA [EXPRESSIVE INTERNET ARCHITECTURE, 2011].*

Observe na **figura 3.8** que o *eXpressive Internet Protocol* (XIP) suporta a comunicação entre os modelos essenciais (*host*, conteúdo e serviço), e, que o XIA trata-se de uma arquitetura flexível, podendo ainda suportar a agregação de novos modelos. Logo acima destes, existe uma ampulheta, que por meio da sua "cintura estreita" generaliza todas as funções chave, incluindo o acesso a objetos (de serviços, hosts e conteúdos, por exemplo), a interação entre as partes interessadas (usuários e provedores de serviços de internet, por exemplo) e gestão de confiança. Isso ajudará a interoperabilidade de todos os níveis do sistema, não apenas o encaminhamento de pacotes. Um resultado desta cintura estreita é que, como a Internet atual, o XIA fornece a abstração de um conjunto de redes individuais, ou seja, uma rede na qual todos os modelos essenciais estão conectados [ANAND, et. al., 2011].

Por meio da identificação de um objeto, é possível verificar algumas propriedades de segurança sem que seja necessário utilizar bases de dados externas [ANAND, et. al., 2011]. Desta forma, é possível, por exemplo, recuperar um conteúdo apenas utilizando seu CID, sendo esta recuperação desacoplada de um *host,* serviço ou endereço específico da rede. Assim, um receptor que conhece a

descrição de um conteúdo, pode recuperá-lo de qualquer lugar e ainda verificar se recebeu os dados corretos.

Apenas a segurança intrínseca nos identificadores não garante uma operação confiável na rede. Faz-se necessário utilizar mecanismos confiáveis (sistemas de publicação, transferência de conteúdo e sistemas de resolução de nomes), que permitirão aos usuários obter corretamente os identificadores referentes ao modelo essencial que desejam acessar. Devem existir ainda mecanismos que garantam a disponibilidade da rede, mesmo sob ataques, criando, portanto, uma rede resistente e confiável. No entanto, até o momento, os mecanismos citados não foram discriminados e divulgados pelos projetistas da arquitetura [ANAND, et. al., 2011].

Devido à grande diversidade de entidades existentes na Internet XIA, será necessário desenvolver uma interface apropriada, permitindo, por exemplo, o usuário identificar não apenas um conteúdo ou sua origem, mas também qual foi a rota delineada até que o conteúdo chegasse ao seu destino e quais serviços foram utilizados neste processo [ANAND, et. al., 2011].

Por se tratar de um projeto recente, elementos adicionais ainda estão sendo estudados e gradativamente divulgados para o público em geral.

3.4 Análise Comparativa

Nesta seção são comparadas e discutidas as características mais relevantes entre os trabalhos relacionados. A **tabela 1** sintetiza as iniciativas apresentadas e detalha como cada uma delas acolhe os princípios da Computação Orientada a Serviços.

Tabela 1: *Comparação de Abordagens.*

Princípios	**CASCADAS**	**BIONETS**	**XIA**
Descrição	GN abrange descrição semântica.	Semântica.	Nomes legíveis.
Identificação	Carece de identificação de serviços.	Carece de identificação de serviços.	Provê a identificação única das entidades de serviços, conteúdos e hosts.
Descoberta	Seleção de serviços usando GN/GA.	Não identificado.	Baseada na resolução de nomes para identificadores.
Composição	Agregação/diferenciação dinâmica, contextualizada.	Dinâmica, genótipo de serviços.	Dinâmica baseada em identificadores.
Localização transparente	Desacoplamento via publicação/assinatura de eventos.	Não identificado.	Desacoplamento via servidor de *lookup*.
Negociação	Direta usando GN/GA.	Não identificado.	Não possui.
Contratação	Direta usando agentes DIET.	Não identificado.	Não possui.
Confiança	Gestão de confiança e reputação.	Gestão de confiança e reputação.	Gestão de confiança.
Mobilidade	Nativa.	Nativa.	Nativa.
Adaptabilidade	*-*awareness*, rede de conhecimento, mudança de planos de ação nos serviços.	Sensoriamento do ambiente, *fitness* aos desejos dos usuários.	Manual. *Context-awareness*.
Autonomicidade	Amplo suporte de funcionalidades auto-*.	Autogeração de serviços (autocatálise).	Manual.
Evolução	Autonômica. Adequação de planos de ações a mudanças de contexto e objetivos.	Autonômica. Inspirada em algoritmos genéticos. Guiada pelos usuários e recursos.	Manual.
Interoperabilidade com Redes IP	Não IP.	IP e não IP.	IP.

Com relação à descrição dos serviços, no CASCADAS, o protocolo *Goal Needed* (GN) descreve semanticamente os serviços necessários em um dado momento da composição dinâmica. No BIONETS, as ontologias são semanticamente relacionadas e uma base de conhecimento global codifica as ontologias e os mapeamentos semânticos. No XIA, a divulgação se dá por meio da escolha de nomes legíveis de cada serviço.

Quanto à identificação dos serviços, somente a iniciativa XIA contempla este aspecto. A identificação única é importante por diversos motivos, como por exemplo, rastreabilidade, melhoria da segurança e por identificar sem repúdio a autoria de comportamentos ilícitos.

No que diz respeito à descoberta de parceiros para a composição de serviços, o CASCADAS utiliza o protocolo GN/GA e considera o contexto exato do serviço desejado. No XIA, a descoberta ocorre apenas por meio de nomes legíveis. Um aspecto comum a estas abordagens é que elas utilizam *software* distribuído para este fim. O ideal é que a descoberta de serviços pudesse usar os mesmos suportes arquiteturais que a descoberta de conteúdos. Isto elimina a sobreposição desnecessária de funcionalidades.

Com relação à composição dinâmica dos serviços, o CASCADAS permite a agregação e diferenciação semântica dos serviços de forma autonômica, sem interferência humana. O BIONETS, além de suportar a composição dinâmica autonômica, permite ainda, a seleção genética dos melhores candidatos a uma dada composição. No XIA, a composição dinâmica é possível através do relacionamento entre SIDs. No entanto, não existe suporte autonômico para isto.

Quanto à localização transparente, é importante para suportar a continuidade dos serviços no caso de eventos de mobilidade de substrato (terminal real ou virtual) e/ou dos próprios serviços (de uma máquina para outra). No CASCADAS este desacoplamento é feito utilizando um mecanismo de publicação e assinatura de eventos distribuídos. No XIA o desacoplamento é natural, ou seja, a arquitetura naturalmente desacopla identificadores de localizadores. Neste caso, existe um servidor de *lookup* que realiza dinamicamente o mapeamento entre identificadores e localizadores.

Quanto à negociação e a contratação, o CASCADAS usa o protocolo GN/GA. Um detalhe interessante é que a negociação é feita diretamente pelos serviços de forma distribuída, sem qualquer entidade centralizadora. Isto é fundamental para o suporte a auto-organização. A contratação usa um ambiente de comunicação orientada a conexão entre agentes do framework DIET. No XIA não existe mecanismos implementados para a negociação e contratação de serviços.

Com relação à confiança na prestação de serviços, todas as abordagens oferecem este suporte. As abordagens CASCADAS e BIONETS incluem também um sistema de gestão de reputação de serviços.

A mobilidade é nativa em todas as abordagens. O suporte a mobilidade generalizada é fundamental em redes futuras, pois cada vez mais os usuários estão utilizando terminais móveis. Espera-se que uma arquitetura de serviços neste contexto suporte a mobilidade de serviços, terminais reais ou virtuais, pessoas e até mesmo redes, tudo isso sem reduzir a disponibilidade dos serviços prestados. Soluções como a abordagem XIA que realiza o desacoplamento de identificadores e localizadores, tem se destacado no suporte à mobilidade generalizada.

Nos quesitos adaptabilidade e autonomicidade, o CASCADAS oferece *context-situation* e *self-awareness*. A ciência da situação caracteriza o ambiente externo a um serviço, enquanto a ciência do *self* (eu) caracteriza o autoconhecimento do serviço sobre suas próprias competências, estado, etc. No CASCADAS, os serviços podem alterar seus planos de ações em função de mudanças no estado interno, do ambiente ou do contexto. Isto é, o serviço evolui para melhor se adaptar ao ambiente, as necessidades do cliente e as suas capacidades internas. Além disso, o CASCADAS oferece amplo conjunto de propriedades auto-*, tais como auto-organização, autoconfiguração, autoproteção, auto-otimização, autocontextualização e autogerenciamento.

No BIONETS, os nós T realizam o sensoriamento da infraestrutura de suporte aos serviços, permitindo que os serviços evoluam em função do estado desses (*infrastructure-awareness*). Também, é feita uma seleção dos serviços que mais se adaptam ao estado do ambiente e as necessidades dos clientes (*user-awareness*). O BIONETS oferece ainda um conjunto de propriedades auto-* e uma funcionalidade bastante inovadora: a chamada autocatálise ou geração espontânea de serviços.

No XIA, não existem mecanismos implementados para a adaptabilidade ou autonomicidade dos serviços. Isto é, não existe algum mecanismo explícito que permita que os serviços se adaptem às mudanças no ambiente ou a outros eventos. Existe, entretanto, a possibilidade de ciência dos nós de rede (*host-awareness*), de ciência aos conteúdos trocados (*content-awarenes*) e de ciência aos domínios pertencentes (*domain-awareness*). Mas, para tirar proveito dessas ciências, novas funcionalidades terão que ser incorporadas ao XIA.

No quesito evolução, o CASCADAS e o BIONETS oferecem evolução autonômica. Isto é, a arquitetura é capaz de evoluir com o mínimo de interferência humana. Observe que isto não significa que os humanos não façam parte desta evolução. Eles fazem, definindo os objetivos, regras, condições de contorno, negócios, etc. Quer dizer que os humanos não precisam gastar tempo excessivo de operação para fazer com que os serviços evoluam. Na atual versão do XIA, a evolução dos serviços ainda é totalmente dependente das ações humanas na arquitetura.

Por fim, tem-se a interoperabilidade com as redes IP. Apenas os projetos BIONETS e XIA contemplam este aspecto. No BIONETS é possível estabelecer a comunicação com ambientes IP utilizando um AP. O XIA utiliza o *eXpressive Internet Protocol* (XIP), protocolo de camada de rede baseado no IP e proposto para substituí-lo.

3.5 Considerações Finais

Embora já tenha ocorrido um grande avanço na direção de trazer a economia de serviços para a Internet, muitos princípios da abordagem SOA ainda continuam apenas parcialmente atendidos pelas propostas apresentadas. Nenhuma das iniciativas discutidas nesse capítulo atende sozinha a todos os atributos do SOA, tampouco aos desafios ainda presentes na Internet de Serviços.

As iniciativas bioinspiradas, CASCADAS e BIONETS surpreendem na questão da auto-organização semântica, gerenciamento de serviços, autonomicidade, adaptabilidade e evolução. Mas, as necessidades dos negócios e dos usuários carecem de maior proximidade, além de não suportarem o desacoplamento de identificadores e localizadores. Por fim, a abordagem XIA possui suporte a este desacoplamento e inova na questão das entidades expressarem as suas necessidades a rede. Mas, carece dos demais recursos existentes nas soluções bioinspiradas e evolucionárias advindas do SOA.

Assim, a principal conclusão é que somente abordagens integradoras, mais amplas, mas restritas a essência do necessário (simplicidade), poderão avançar

no sentido a satisfazer o real potencial da Internet de Serviços. Ou seja, existe uma ótima oportunidade de pesquisa e desenvolvimento no atual cenário, principalmente quando se considera a importância do terceiro setor na economia: serviços.

Capítulo 4

Sistemas da Arquitetura

Este capítulo é dividido em seções que abordam conceitos e características de parte dos sistemas (*DS – Domain System, PSS – Public Subscribe System, DHTS – Distributed Hash Table System, GIRS – Generic Indirection Resolution System, SDS – Search and Discovery System*) da arquitetura que está sendo projetada pelo orientador deste trabalho, e que inclui os resultados de [MARTINS, 2011] e [VAZ, 2011]. Eles serão utilizados na proposta do Sistema Genesis – principal contribuição desta dissertação – apresentada no capítulo 5.

4.1 DS – Domain System

Um domínio é definido como um grupo de contas e recursos de rede que compartilham um mesmo banco de dados de diretórios e um conjunto de diretivas de segurança, podendo ter relacionamentos de segurança com outros domínios [MICROSOFT, 2011]. Relacionando esta definição com a arquitetura proposta, um domínio pode ser definido como um conjunto de entidades e sistemas com relações de confiança bem definidas, onde todas as publicações são organizadas com base em ontologias. As entidades habitantes do domínio podem estabelecer relações de competição e cooperação entre elas, bem como com entidades de outros domínios.

Neste ambiente, cada domínio possui seu representante ou *Domain System* (DS), que é responsável por representar os interesses do domínio perante entidades DS de outros domínios, conduzindo os processos de busca e seleção de

domínios parceiros; gerenciando interações de entidades do seu domínio com domínios parceiros, autorizando ou negando a replicação de consultas e oportunidades de contrato; e por fim, estabelecendo contratos de cooperação com entidades DS de outros domínios, o que possibilita criar um domínio de domínios e assim por diante.

Os domínios de alto nível podem vir a representar os interesses dos domínios de baixo nível (domínios filhos de domínios), situação similar à segmentação de redes da arquitetura centrada em *hosts*. A estrutura em domínios hierárquicos permite ainda criar entidades compostas por entidades de múltiplos domínios parceiros, como, por exemplo, um serviço composto por serviços que foram publicados em outros domínios.

4.2 PSS – Public Subscribe System

O *Public Subscribe System* (PSS) permite às entidades de um domínio ou de domínios parceiros realizarem publicações e assinaturas de conteúdos e serviços, bem como receberem notificações umas das outras. Ele implementa o paradigma publica/assina. A publicação é realizada quando é registrada a identificação única do conteúdo ou serviço no GIRS (*Generic Indirection Resolution System*) (discutido na seção 4.4), e a assinatura, é realizada quando alguma entidade interessada solicita ao GIRS algum conteúdo ou serviço publicado. Desta forma, a localização e contratação de um serviço ou localização e acesso de um conteúdo só é possível depois que seu objeto tenha sido publicado. Neste trabalho, utilizam-se os comandos ***Pub***, para realizar a publicação, ***Sub*** para realizar a assinatura e ***Notify*** para realizar a notificação opcional de publicações/assinaturas feitas.

As publicações e assinaturas podem ser realizadas de duas formas: com notificação ou sem notificação. A primeira ocorre quando é necessário que o PSS notifique a outra entidade a respeito da publicação ou assinatura de algum mapeamento, e possui a seguinte estrutura: **Pub/Sub (Chave; Notify = ID-Notificação; <Mapeamento>, Tipo)**, onde a Chave é a identificação da entidade que está publicando ou assinando no PSS e receberá de volta, uma mensagem

Acknowledge (***Ack***) reconhecendo que o mapeamento foi publicado ou assinado com sucesso; o campo ID-Notificação é uma identificação da entidade que será notificada pelo PSS a respeito da publicação ou assinatura do referido mapeamento; o Tipo é o número do mapeamento e possibilita diferenciar publicações e assinaturas de tipos diferentes de entidades habitantes.

Já as publicações ou assinaturas sem notificação são aquelas em que não existe a necessidade de notificar a outra entidade, possuindo a seguinte estrutura: **Pub/Sub (Chave; <Mapeamento>; Tipo).**

Considere o cenário onde uma entidade publicadora (Publicador) realiza a publicação de um conteúdo ou serviço que será assinado por uma entidade assinante (Assinante). Ao realizar a publicação o Publicador recebe um ACK informando que a publicação foi bem sucedida ou um *negative-acknowledge* (NACK), caso contrário. Dado que a publicação foi realizada com sucesso, o Assinante realiza uma busca por objetos publicados, seleciona o resultado e ao assinar a publicação, o PSS notifica o Publicador de que seu conteúdo ou serviço foi assinado. Observe que o PSS só autoriza a assinatura se o Publicador tiver autorizado o Assinante previamente. Em seguida, é iniciado o processo de negociação entre as entidades envolvidas, e assim que chegarem a um acordo, o fornecimento do serviço ou conteúdo requerido será permitido.

O sistema apresentado nesta seção, PSS, está diretamente envolvido com a proposta deste trabalho, uma vez que possibilita: (i) implementar um serviço de encontros baseado na confiança, segurança e autenticação, (ii) implementar mensagens de notificação e (iii) realizar mapeamentos GIRS, sistema que será apresentado na seção 4.4.

4.3 DHTS - Distributed Hash Table System

As *Distributed Hash Tables* (DHTs) são uma classe de sistemas distribuídos descentralizados que prestam um serviço de pesquisa baseado em Tabelas *hash* distribuídas. Pares (*chave, valor*) são armazenados nos DHTSs, podendo qualquer nó participante recuperar o valor associado a uma dada chave. A

responsabilidade de manter o mapeamento entre as chaves e valores é distribuída entre os nós. Desta forma, o DHTS pode escalar para um número extremamente grande de nós, bem como ser tolerante a falhas, lidando com conectividade intermitente, como a entrada ou saída dos nós da rede, por exemplo [DISTRIBUTED HASH TABLE, 2011] e [LIU, et. al., 2009].

Como apoio à escalabilidade, a responsabilidade pela manutenção das tabelas de mapeamento é distribuída entre os nós participantes, não sendo necessário que estes conheçam todos os seus vizinhos, mas sim que tenham uma pequena tabela de roteamento contendo os identificadores e localizadores de alguns dos seus pares vizinhos [CIRANI & VELTRI, 2007]. Desta forma, o alcance a um nó qualquer é conseguido com um número relativamente pequeno de saltos, variando conforme o algoritmo utilizado (*Chord* [STOICA, et. al., 2003], *Kademlia* [MAYMOUNKOV & MAZIERES, 2002], *Tapestry* [ZHAO, et. al., 2004] *ou Pastry* [ROWSTRON & DRUSCHEL, 2004], por exemplo).

O DHTS tem basicamente a mesma interface de uma Tabela *hash* normal, mas atuam por meio de uma rede distribuída. A **figura 4.1** ilustra a ideia de um ambiente DHTS.

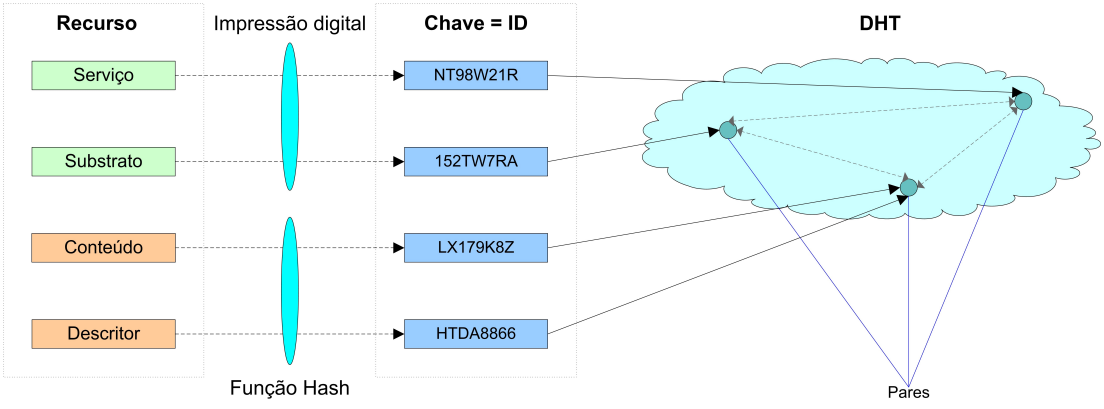

Figura 4.1: *Ambiente DHT.*

Observe na **figura 4.1**, que foi utilizada uma função *hash* para identificar o recurso de forma única e mapear estas chaves para seus respectivos valores. A chave de um recurso (conteúdo, serviço ou *host*) geralmente é formada pelo *hashing* do próprio recurso em si ou de uma impressão digital, enquanto que o valor é um

dado associado ao recurso (como alguns metadados e/ou endereço de localização onde o recurso pode ser encontrado) [CIRANI & VELTRI, 2007].

A **figura 4.2** ilustra de forma generalizada o funcionamento de uma DHT.

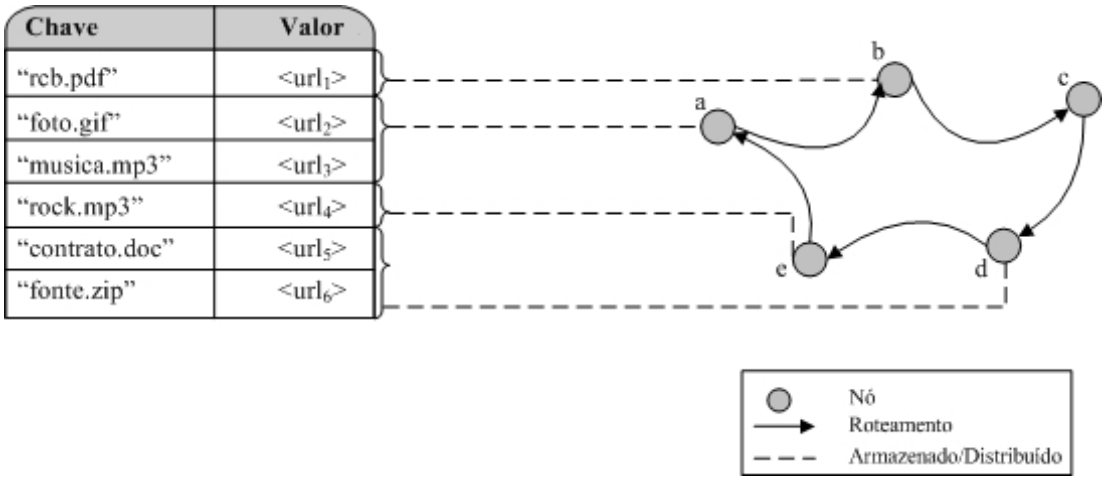

Figura 4.2: *Mapeamentos de chaves e valores distribuídos entre nós DHTs [GHODSI, 2006].*

Considere a **figura 4.2**, onde a título de exemplo o mapeamento é realizado entre os nomes de recursos que são utilizados como chaves em consultas DHTs e seus respectivos valores URL (*Uniform Resource Locator*). Os registros DHTs (chave, valor) 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 um sistema de pesquisa faça uma solicitação ao nó *a* para descobrir a localização atual do arquivo "rcb.pdf". O nó *a* irá rotear o pedido ao nó *b*, que por sua vez irá informar ao nó *a* que pode atender ao pedido, já que, conhece a URL associada à chave "rcb.pdf".

É importante ressaltar que cada operação de roteamento realizada garante que uma cópia local do documento seja mantida nos nós intermediários. Desta forma, quando um par solicita o documento de um sistema *peer-to-peer*[6] (P2P), a requisição irá até o par que possui a chave mais parecida com a chave do documento. Este processo continua até que uma cópia do documento seja encontrada. Então, o documento é transferido ao par que originou a solicitação, enquanto cada par que

[6] ***Peer-to-peer*** refere-se a uma classe de sistemas e aplicações que utilizam recursos distribuídos para desempenhar uma função de forma descentralizada [MILOJICIC, et. al., 2003].

participou do roteamento continuará com uma cópia local do documento [STOICA, et. al., 2003].

4.4 GIRS - Generic Indirection Resolution System

O *Generic Indirection Resolution System* (GIRS) é um sistema proposto por [MARTINS, 2011] para tratar indireções[7] utilizadas em propostas de Internet do futuro, em especial nas redes centradas em informação, uma proposta que trata o principal objetivo das redes: troca e processamento de informação, independente da sua localização [JACOBSON, et. al., 2009]. O GIRS pode ser usado para mapear de forma inequívoca a chave de um objeto para o identificador do nó responsável pelo objeto desejado, além de possibilitar relacionar descritores de algumas entidades com identificadores de outras.

Para que este sistema funcione corretamente, todas as entidades (reais e virtuais) devem ser identificadas de forma única; receber um nome legível em linguagem natural, de modo a facilitar o uso do sistema por parte das pessoas; possuir um ou mais objetos descritores para descrevê-las, permitindo usuários e sistemas localizarem semanticamente as entidades desejadas, e por fim ter um localizador que represente a sua localização na topologia da rede. Estas características de uma mesma entidade ou de entidades distintas se relacionam através de mapeamentos para a localização e recuperação de objetos de conteúdo [MARTINS, 2011]. A **figura 4.3** ilustra a estrutura do GIRS.

[7] **Indireção** pode ser usada para referenciar uma entidade usando um nome ou um rótulo em vez de o valor em si.

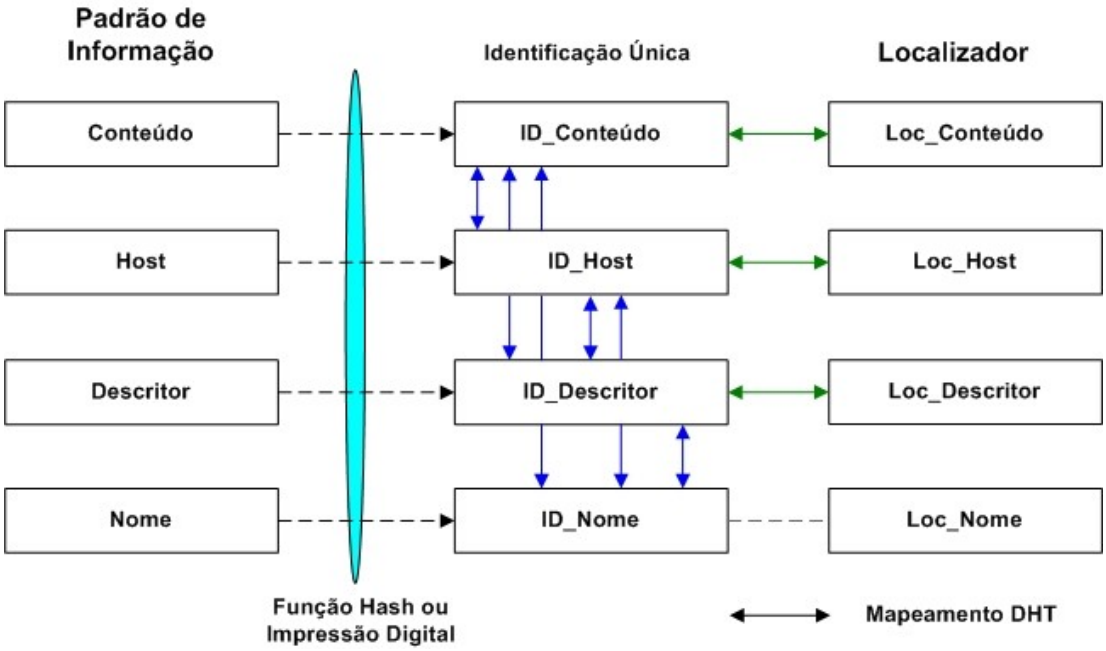

Figura 4.3: *Estrutura do GIRS [MARTINS, 2011].*

O padrão de *bits* de um conteúdo digital, como por exemplo, uma imagem, é estatisticamente único, ou seja, a probabilidade de que outra imagem que não seja uma cópia ter o mesmo padrão de *bits* é estatisticamente desprezível. Desta forma, é possível usar este padrão para criar uma identificação única para este objeto. Para cada entidade identificada unicamente, pode-se associar um descritor, que a descreve para uso geral. É possível ainda, associar um mesmo descritor a mais de uma entidade, no caso de existirem variações de formato e por fim, associar um nome a este objeto [MARTINS, 2011].

A **figura 4.3** ilustra ainda a possibilidade de usar esta mesma sequência de indireções para *hosts*. Afinal, cada *host* tem um padrão único, que pode ser obtido a partir de uma combinação única de características da máquina, i.e. sua impressão digital. A partir deste padrão estatisticamente único, é possível criar um identificador único para este *host*, relacionar seu identificador com descritores, e ainda dar um nome legível a este *host*, como é feito na atual arquitetura centrada em *hosts* [MARTINS, 2011].

Neste trabalho, foram realizadas algumas alterações na estrutura do GIRS, sendo modificado o padrão de informação para entidades habitantes da arquitetura e removido o item Localizador. Isto foi feito, pois as entidades e recursos

disponíveis na rede foram estruturadas com base em ontologias publicadas, que representam os significados dos termos utilizados. Por exemplo, uma ontologia sobre "redes de dados" poderia modelar "servidor" como um computador com alta capacidade de processamento e armazenamento que tem a função de disponibilizar um serviço específico a uma rede, enquanto uma ontologia de "concurso público" poderia modelar seu significado como um servidor público, que é a pessoa investida em cargo ou emprego público.

É importante ressaltar que as ontologias não servem apenas como vocabulário de comunicação entre as entidades e sistemas, mas também para a definição e organização adequada de conceitos, relações e restrições. Desta forma, a ontologia de um domínio deve ser bem detalhada de forma a garantir precisão na classificação e identificação de suas entidades habitantes. A **figura 4.4** ilustra a estrutura do GIRS conforme o escopo deste trabalho.

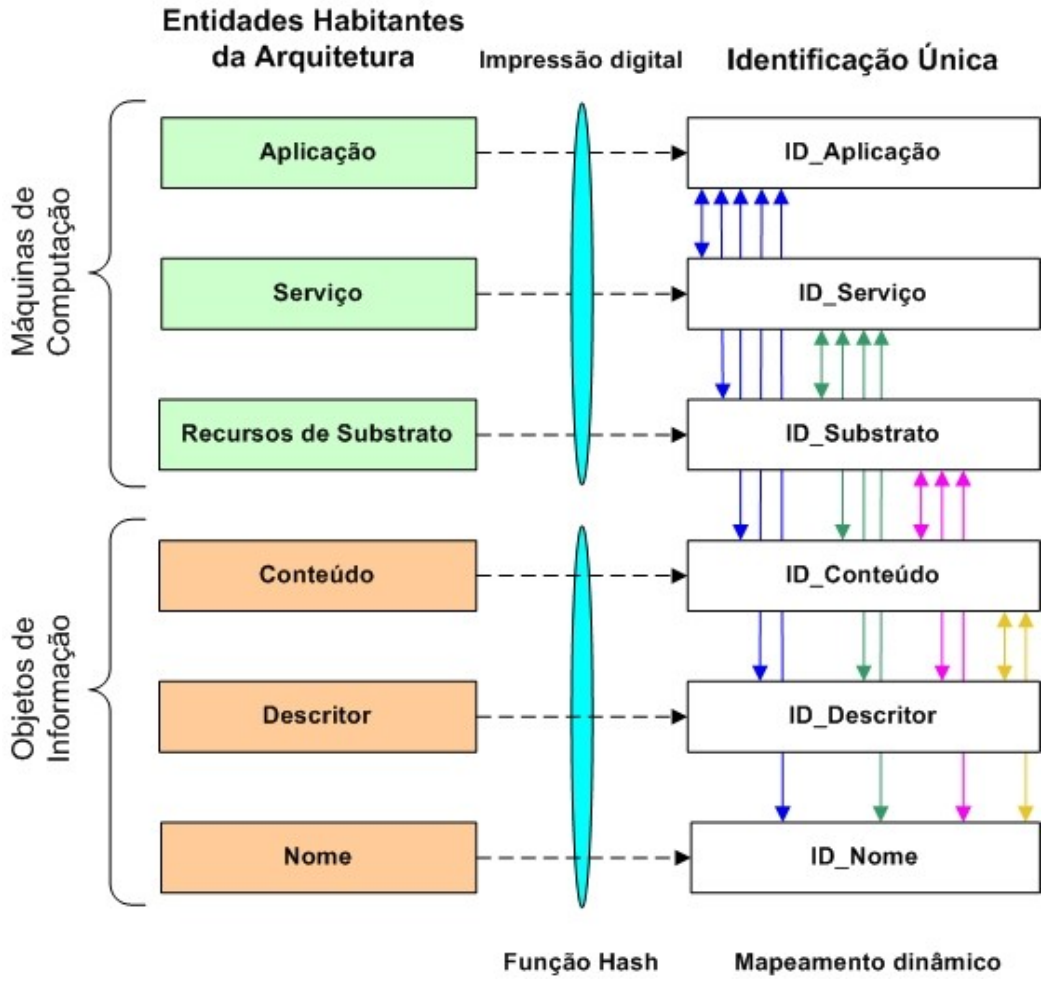

Figura 4.4: *Nova estrutura do GIRS.*

A cada serviço pode-se associar um recurso de substrato, que representa o local onde ele estará executando ou armazenado; um conteúdo, que define, por exemplo, as condições para sua contratação ou fornecimento, sendo possível ainda, associar um descritor e nome a este serviço. O mesmo procedimento poderá ser realizado com a aplicação, podendo esta ser resultado da agregação de vários serviços, um *cluster* de serviços. Um recurso de substrato pode ser também uma interface de rede.

Perceba que o GIRS é um sistema que deverá ser frequentemente utilizado por todas as entidades habitantes do domínio e/ou de domínios parceiros, atuando como um catálogo que mapeia identificadores distribuídos as suas respectivas entidades. Por meio deste sistema é possível ainda determinar o relacionamento entre identificadores de diversas entidades, criando assim uma cadeia de rastreabilidade, contexto e semântica baseada em uma estrutura ontológica.

4.4.1 Mecanismos de Resolução

Na proposta apresentada por [MARTINS, 2011], o mecanismo de resolução de indireções foi dividido em três módulos: (*i*) um *Search Engine* (SE), que fornece pesquisa e recuperação de entidades com base na semântica, metadados e atributos destas entidades, (*ii*) um *Name Resolution System* (NRS) responsável por mapear identificadores em localizadores de objetos de informação e (*iii*) um mecanismo para mapear identificadores e localizadores de *hosts* (*Split Host*).

Neste trabalho, o GIRS é composto somente pelo módulo NRS, sendo responsável apenas pela resolução de indireções. O módulo SE foi modificado e será apresentado de forma independente na seção 4.5, por meio do *Search and Discovery System* (SDS). A separação destes sistemas foi realizada com o intuito de simplificar e descentralizar a arquitetura proposta, bem como, garantir que todas as comunicações entre os sistemas sejam realizadas por meio do PSS. Por fim, o módulo *Split Host* foi retirado do escopo desta proposta, uma vez que o acesso aos objetos é realizado por meio de ontologias da rede e não mais de localizadores.

4.4.2 Estrutura de Mapeamento para o GIRS

Os atributos (nomes, identificadores e descritores) das entidades habitantes da arquitetura proposta são relacionados por meio de mapeamentos registrados no DHTS. A **figura 4.5** ilustra a estrutura do mapeamento para o GIRS.

Chave (512 bits) (Hash da fonte do mapeamento)
Tipo do Mapeamento
Lista de Valores Destino do Mapeamento
Tempo de Vida

Figura 4.5: *Estrutura dos mapeamentos para o GIRS [MARTINS, 2011].*

Conforme ilustrado na **figura 4.5**, a chave é obtida por meio de uma função *hash* de 512 *bits* do objeto fonte do mapeamento ou através de uma impressão digital das máquinas computacionais. Ela representa o valor de entrada para as pesquisas de valor associado no DHTS. O tipo de mapeamento representa o estilo de resolução entre as entidades envolvidas na pesquisa. A lista de valores corresponde aos valores retornados de um mapeamento associado a uma dada chave. Por fim, o tempo de vida é utilizado para determinar a expiração e a atualização dos mapeamentos.

4.4.3 Estrutura do Descritor

O descritor é o objeto que contém as especificações do habitante que ele descreve. Estas especificações são constituídas por um conjunto de atributos que podem genericamente representar: um nome do objeto (“criptografia”, “autenticação compartilhada” e “fotos Vancouver”, por exemplo), informações de classificação do objeto (serviço, vídeo, som, documento de texto ou imagem, por exemplo), formato do arquivo (EXE, MPEG, MP3 ou PDF, por exemplo), atributos do mundo real fornecidos por etiquetas RFID (*Radio Frequency Identification*); *chips* NFC (*Near Field Communication*) ou WSANs (*Wireless Sensor and Actuator Networks*) para monitoramento de plantas e casas, por exemplo.

Perceba que não existe um padrão definido para as especificações dos descritores. Contudo, a preferência é usar linguagem natural, para permitir que as entidades (usuários, aplicações e sistemas, por exemplo) selecionem o padrão desejado com base na semântica fornecida pelos descritores. A **figura 4.6** ilustra uma estrutura genérica para estes descritores.

ID
Lista de Identificadores de Padrões Relacionados
Lista de Atributos
Localizador do Descritor
Tempo de Vida – *TTL* (*Time to Live*)

Figura 4.6: *Estrutura do descritor para o GIRS [MARTINS, 2011].*

Conforme apresentado na **figura 4.6**, todo descritor contém um identificador (ID), gerado através do seu padrão binário; um identificador ou uma lista de identificadores de objetos descritos (Lista de Identificadores de Padrões Relacionados); uma relação de atributos relacionados a este objeto (Lista de Atributos) para descrever o padrão ou uma série de padrões relacionados; e o localizador ou identificador do substrato (opcional) que armazena este objeto descritor (Localizador do Descritor); e um tempo de vida para a prática de atualização e expiração (*TTL -Time to Live*).

4.4.4 Ajuste de Nomes

Os identificadores gerados através de funções *hash* não são legíveis e não possuem informações de semântica e de contexto das entidades da arquitetura que foram identificadas. Faz-se necessário então, identificar legivelmente as entidades habitantes (serviços, sistemas e conteúdos, por exemplo), de forma que estes nomes possam auxiliar o processo de busca e localização dos objetos [MARTINS, 2011]. A existência de diferentes serviços e conteúdos publicados pode gerar resultados homônimos. Desta forma, exige-se que os nomes criados tenham um alto grau semântico, como por exemplo, "servidor de arquivos do departamento de tecnologia da empresa ABCD".

Para evitar problemas de nomeação, propõe-se que os nomes passem por uma codificação de fonte e um ajuste de tamanho mínimo. A codificação é utilizada na geração de identificadores para os nomes, padronizando a nomeação das entidades existentes na arquitetura. O ajuste padroniza o nome em um tamanho mínimo definido, evitando assim, o uso de nomes muito pequenos antes de passar pela função *Hash*. A **figura 4.7** ilustra o processo de criação de nomes, desenvolvido em [MARTINS, 2011].

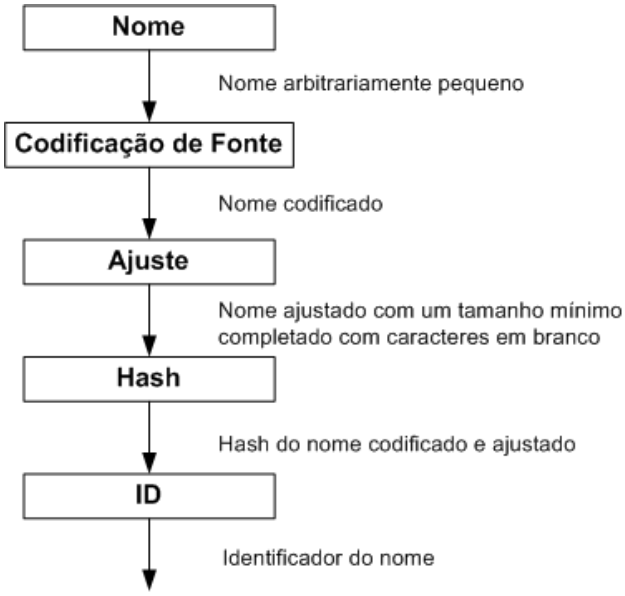

Figura 4.7: *Codificação de fonte e ajuste para nomes arbitrariamente pequenos [MARTINS, 2011].*

4.5 *SDS* - Search and Discovery System

O *Search and Discovery System* (SDS) possibilita realizar consultas e recuperar entidades, considerando não apenas padrões estruturais e sintáticos, mas também padrões semânticos. A busca semântica visa melhorar a precisão da busca por compreender a intenção da entidade pesquisadora e o significado contextual da descrição informada, retornando o resultado solicitado ao invés de uma lista de resultados baseada simplesmente em palavras chave relacionadas.

Iniciada uma pesquisa, o sistema por meio das diversas ontologias definidas no domínio realiza uma análise semântica para entender o que a entidade está procurando, caracterizando assim a desambiguação. Quando um termo é ambíguo, este pode ter vários significados (por exemplo, se considerarmos o termo

"rede", este pode ser entendido como "rede de pesca", "rede de computadores", "rede de telecomunicações", entre outros), o processo de desambiguação é realizado quando o significado mais provável é escolhido a partir de todos esses possíveis. A **figura 4.8** ilustra uma pesquisa realizada por meio do SDS.

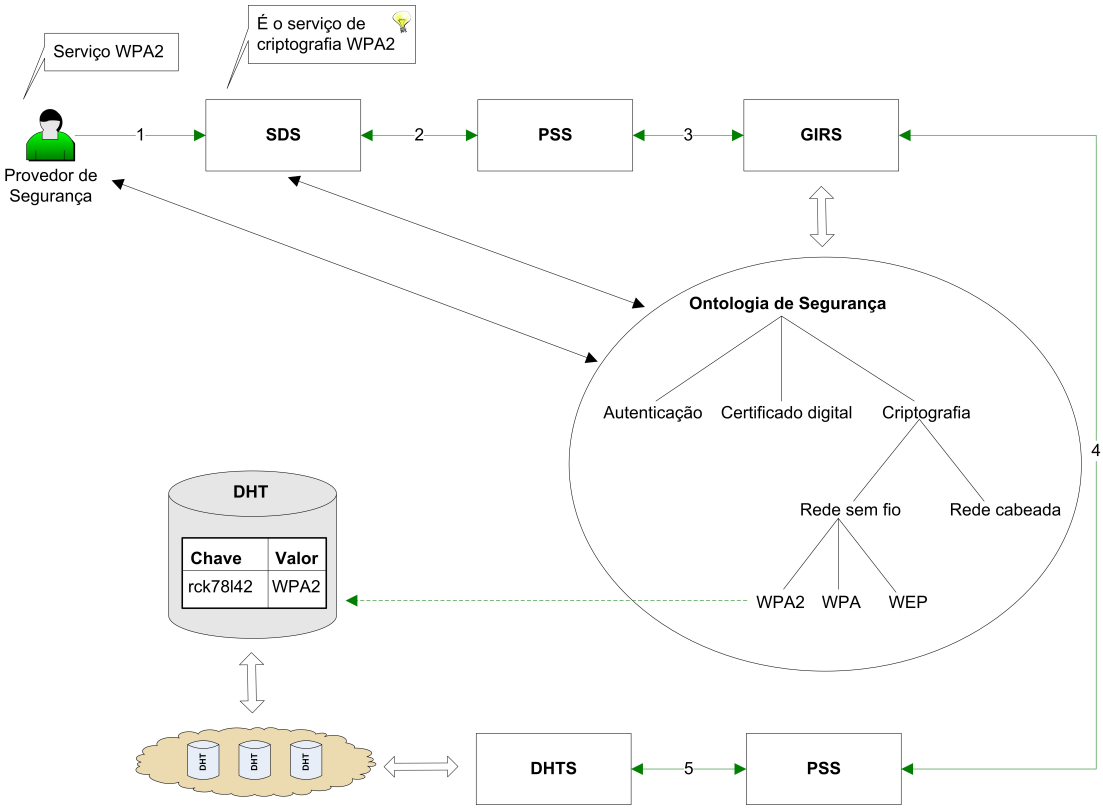

Figura 4.8: *Sistema de Descoberta e Busca.*

Considere que uma entidade pesquisadora (Provedor de Segurança) deseja verificar se existe o serviço "*WPA2"* (*Wi-Fi Protected Access 2*) publicado no domínio a qual pertence. Neste, o SDS e a entidade em questão compartilham a "ontologia de segurança de redes" e ambos têm em suas bases de conhecimento o fato de que "*WPA2"* é um serviço de criptografia para redes *wireless*. Desta forma, o SDS consegue compreender a descrição informada pela entidade, realizar a busca semântica e retornar o resultado mais condizente.

Perceba que o SDS possui conhecimento do ambiente em que está inserido, possibilitando a entidade pesquisadora especificar seu interesse em menos detalhes no momento da consulta, caso compartilhem a mesma ontologia. Caso contrário, a entidade deverá especificar seu interesse em mais detalhes.

É importante ressaltar que é possível descobrir e consultar apenas entidades que tenham sido previamente publicadas no PSS e naturalmente no GIRS do seu domínio ou de domínios parceiros.

4.6 Considerações Finais

Este capítulo abordou os sistemas básicos que serão utilizados na proposta do Sistema Genesis – apresentada no Capítulo 5. Foram apresentados cinco sistemas: (i) O DS – *Domain System*; (ii) O PSS – *Public Subscribe System*; (iii) O DHTS – *Distributed Hash Table System*; (iv) O GIRS – *Generic Indirection Resolution System*, e por fim, (v) O SDS - *Search and Discovery System*. O primeiro sistema é responsável por representar os interesses do domínio perante outros domínios, conduzindo processos de busca e seleção de domínios parceiros e estabelecendo contratos de cooperação. O segundo sistema possibilita as entidades de um domínio ou de domínios parceiros realizarem publicações e assinaturas de conteúdos e serviços, bem como receberem notificações umas das outras. O terceiro sistema é uma classe de sistemas distribuídos que oferecem um serviço de pesquisa baseado em tabelas *hash* distribuídas. O quarto sistema atua como um catálogo de endereços, mapeando de forma inequívoca identificadores distribuídos às suas respectivas entidades. Finalmente, o quinto sistema possibilita as entidades realizar consultas e recuperar conteúdos e serviços, considerando não apenas padrões estruturais e sintáticos, mas também padrões semânticos.

Capítulo 5

Sistema Genesis

Neste capítulo apresenta-se o *Genesis System* (GS), responsável pela instanciação de todos os demais sistemas (DHTS, PSS, GIRS, SDS e DS) de uma proposta de arquitetura para a Internet do Futuro em concepção no INATEL. Além de criar instâncias desses sistemas, o GS também é responsável pela criação de um *Orchestration Broker System* (OBS), de um *Reputation System* (RS) e de um *Compilation and Decompilation System* (CDS), bem como pela criação de entidades virtuais utilizadas em propostas de Internet do Futuro, em especial nas redes centradas em serviço. A arquitetura na qual o GS presta o serviço de instanciação considera que existem somente dois tipos de entidades habitantes. Ou o habitante é uma informação (padrão de ordem que serve a um objetivo), ou é uma computação (processador de informação). A **figura 5.1** ilustra tais entidades habitantes no contexto do Sistema Genesis.

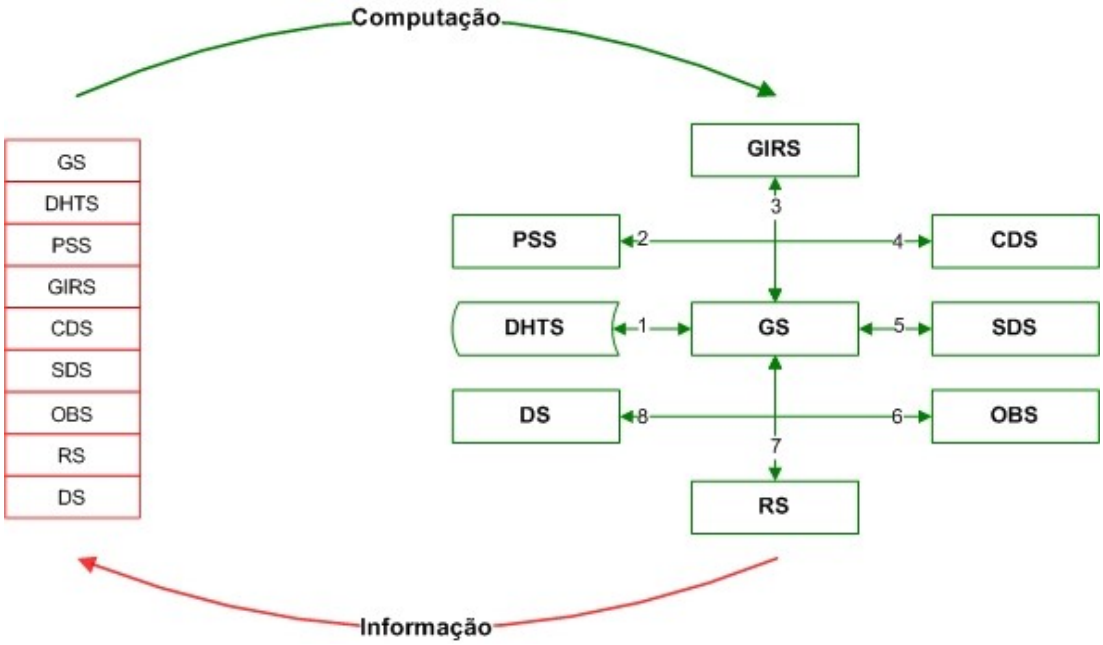

Figura 5.1: *GS – Informação e Computação.*

Note que o GS é potencialmente capaz de salvar o estado da arquitetura virtual, retornando ao estado de informação e vice-versa. O estado de informação pode ser alcançado de duas formas: (*i*) quando os sistemas deixam de ser executados ou (*ii*) quando retornam ao código fonte por meio de um sistema de descompilação, por exemplo. O caminho reverso também é verdadeiro, podendo o estado de computação ser alcançado na primeira forma, quando o sistema ou serviço é novamente executado ou na segunda forma, quando é novamente compilado e executado. A primeira forma pode ser uma opção mais conveniente às empresas de serviços pagos (Microsoft e Apple, por exemplo). Já a segunda forma, pode ser uma opção mais conveniente às empresas provedoras de serviços de código livre (Distribuições Linux, por exemplo).

Observe ainda na **figura 5.1**, que foram instanciados ordenadamente os sistemas básicos da arquitetura que compõem o domínio e em seguida o DS que irá representar os interesses de as entidades do domínio perante outros domínios. Ou seja, primeiro será necessário criar a estrutura do domínio, para que então este possa ser representado.

Na nova arquitetura, todos os sistemas e serviços “vivem” em um ambiente virtual, podendo a criação de novas instâncias de serviços ser realizada de

três formas: manual, autonômica centralizada ou autonômica distribuída. A primeira é quando o próprio usuário desenvolve o serviço por completo ou utiliza a invocação semântica para identificar serviços potenciais que poderão compô-lo. Na segunda, o OBS através da auto-organização semântica compõe novos serviços para atender a alguma demanda específica do serviço do usuário. Auto-organização semântica permite a entidades de serviço identificar serviços fundamentais para compor serviços mais sofisticados. Na terceira, os próprios serviços por meio da auto-organização semântica possuem a capacidade de negociar com outros e se autocomporem de forma totalmente distribuída.

Perceba que além da forma tradicional de criação de serviços (*software*s de prateleira e bancos de dados, por exemplo), existe também a criação por composição que poderá ser alcançada de forma autonômica, ou seja, é realizada por entidades virtuais. Ou, simplesmente por invocação manual, que é realizada pelos próprios usuários.

Generalizando a composição de serviços, esta pode ser realizada de duas formas: (*i*) através da busca semântica direta de descritores de serviços ou (*ii*) por meio da publicação de Oportunidades de Contrato (*Contract Opportunity* - CO). COs são documentos que descrevem todas as condições para a negociação de contratos de serviço, bem como as características que alguma entidade deseja contratar. Na primeira forma, a entidade requerente (usuários, serviços de usuários ou OBS) realiza uma busca semântica direta por descritores de serviços de seu interesse. Se houver resultados disponíveis, estes são retornados para a entidade na forma de descritores e/ou nomes legíveis. Então, com base nestas informações, a entidade elege os serviços que deseja contratar, e em seguida avança para os processos de negociação, contratação e fornecimento dos mesmos. Caso a entidade requerente não encontre algum serviço condizente com a pesquisa ou durante o processo de negociação julgue inviável contratar os serviços elegidos, poderá publicar uma CO, iniciando a segunda forma possível de composição. Uma vez que a CO esteja publicada, as entidades realizam uma busca semântica por conteúdo, sendo retornados os resultados disponíveis. A entidade elege as COs condizentes com sua área de especialização, sendo na sequência realizados os processos de negociação,

contratação, composição e fornecimento do serviço. Observe que a CO também contém informações semânticas sobre a oportunidade de contrato existente.

Concluída as contratações e a composição do novo serviço, este será compilado manualmente ou de forma autonômica por meio do CDS, que será apresentado na subseção 5.1.3. Em seguida, será identificado por algum mecanismo de geração de identificadores únicos, nomeado, descrito e publicado no PSS. Posteriormente, será publicado no GIRS e armazenado no DHTS, possibilitando as entidades do domínio tomar conhecimento de sua existência.

Observe que a criação de um serviço através da agregação de vários outros pode ser resumida como um *script* de vários contratos (estado de informação) ou um *cluster* de vários serviços em execução (estado de computação).

Para que seja possível criar, publicar, assinar, armazenar e contratar serviços e conteúdos é imprescindível que se tenha um ambiente virtualizado. Recursos virtuais são serviços e, por exemplo, no caso de um *upgrade*, estes devem ser pesquisados, negociados e contratados. Neste cenário, um desafio é a criação do primeiro ambiente virtual onde os demais sistemas e serviços da arquitetura serão instanciados. Propõe-se então, a utilização de uma licença temporária baseada em contratos iniciais que independem de controles de admissão prévio, adquirida de forma manual, por exemplo, entre pessoas físicas (representantes de um domínio) e pessoas jurídicas (representantes de empresas provedoras de ambientes virtuais). Esta solução permite contratar a infraestrutura virtual que irá compor o novo domínio no qual os sistemas da arquitetura serão instanciados. Se for do interesse da entidade requerente, essa licença poderá ser prorrogada utilizando-se os sistemas instanciados no próprio ambiente temporário.

O processo apresentado até o momento refere-se ao cenário mais simples, onde as buscas por serviços e a publicação de Oportunidades de Contrato foram realizadas em um único domínio e este dispunha de serviços condizentes. O cenário mais complexo será publicar Oportunidades de Contrato em vários domínios e nenhum deles possuir serviços (nos estados de computação ou informação) condizentes com a publicação. Desta forma, as entidades interessadas na

Oportunidade de Contrato devem formular estratégias de negócios voltadas à cooperação e competição simultaneamente criando condições para atender a demanda, como por exemplo, publicando novas oportunidades de contrato de partes do serviço que deseja compor. A seguir esses sistemas são discutidos em maiores detalhes, incluindo cenários de estabelecimento de contrato.

5.1 GS – Genesis System

Após a instanciação dos sistemas da arquitetura, o GS será responsável por criar as entidades OBS, RS e CDS que serão apresentadas nas subseções 5.1.1, 5.1.2 e 5.1.3 respectivamente.

5.1.1 OBS - Orchestration Broker System

É uma entidade pró-ativa, ciente da situação e do contexto, capaz de aprender com suas experiências e tomar suas próprias decisões com o mínimo de interferência humana.

O *Orchestration Broker System* é constituído por funcionalidades comuns e específicas. A primeira funcionalidade diz respeito à sua atividade primordial, que é auxiliar os serviços de usuários que não são capazes de negociar e contratar serviços, representando seus interesses e realizando sua composição semântica. A segunda funcionalidade refere-se à área de negociação ou especialização associada a cada entidade OBS. A **figura 5.2** ilustra o processo de seleção do OBS conforme funcionalidade.

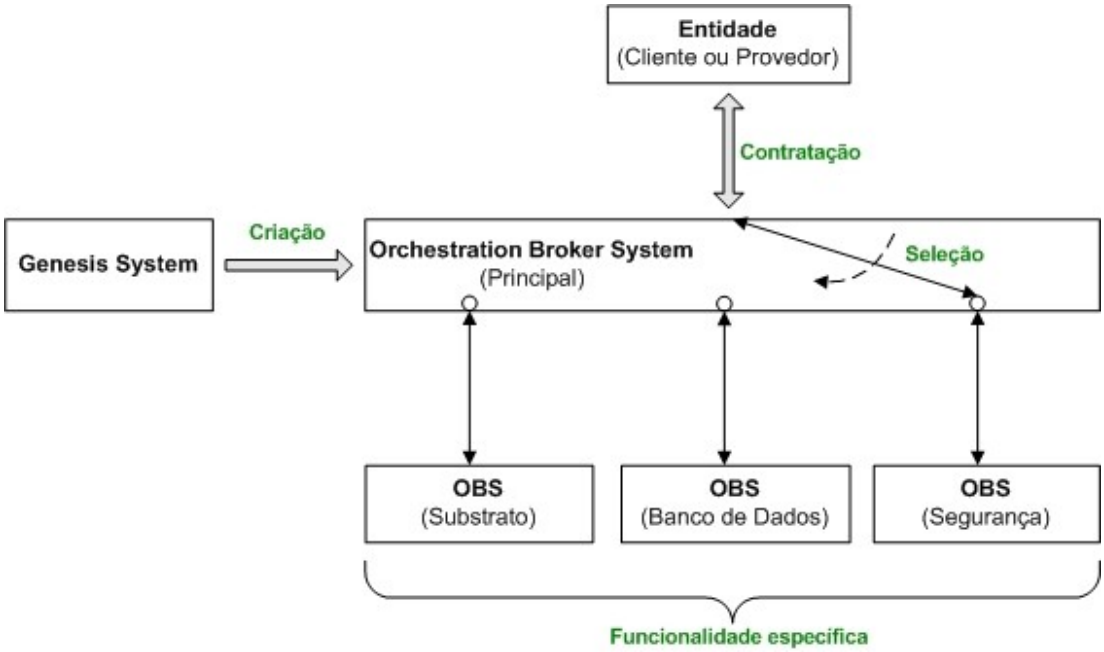

Figura 5.2: *OBS associado a algumas áreas de especialização possíveis.*

Após ser criado pelo GS, o OBS (principal) por meio da estrutura de ontologias da rede mapeia as áreas que precisam ser representadas (segurança, banco de dados e substrato, por exemplo) e então, cria entidades OBS auxiliares para representá-las.

Considerando que o OBS está operando, sua prestação de serviços pode ser contratada de duas formas: (*i*) manual, em que o usuário junto ao OBS define como deverá ser realizada a representação do seu serviço ou (*ii*) autonômica, onde um serviço capaz de negociar contrata o OBS para representar os interesses de outro serviço incapaz de se auto representar, como é o caso por exemplo de serviços no estado de informação orquestrados por outros serviços. Por fim, assim como todos os contratos estabelecidos dentro da arquitetura proposta, este deverá ser formalizado por meio de um SLA.

Dado que o SLA foi estabelecido com sucesso, o OBS deverá ter uma participação ativa na busca por oportunidades de mudança, planejamento e composição de novos serviços, solucionando e antecipando-se aos problemas, e propondo metas que beneficiem os serviços que representa. Para tanto, deverá ter ciência do ambiente em que está inserido e naturalmente das informações que caracterizam determinadas situações.

Neste contexto, a entidade OBS pode se relacionar de maneira cooperativa com todas as entidades da rede e vice-versa. Através da cooperação, o OBS pode representar os interesses de seus contratantes, negociando seu fornecimento ou composição de novos serviços junto à outras entidades da rede. Além da cooperação, o OBS compartilha questões de competição, que podem existir com outras entidades OBS, usuário, ou mesmo com serviços que sejam capazes de negociar seus próprios interesses. Da mesma forma que a competição entre as empresas é um determinante crítico no desenvolvimento de novos produtos e tecnologias, a competição entre as entidades na rede pode proporcionar uma contínua inovação dos serviços.

Por meio das diferentes demandas que possam surgir e sucessivas cooperações e competições com as demais entidades do seu domínio e de domínios parceiros, a entidade OBS aumenta seu nível de conhecimento, tornando-se uma entidade cada vez mais apta a orquestrar serviços semanticamente.

5.1.2 RS – Reputation System

É um sistema que gerencia o *feedback* de entidades (serviços, por exemplo) com relação aos contratos estabelecidos, sendo a reputação de cada entidade atualizada ao final de cada contrato. As entidades contratadas serão avaliadas e receberão notas (méritos e/ou deméritos) que irão compor sua reputação. Assim, quando participarem de novos processos de negociação, o RS poderá ser consultado, possibilitando às entidades requerentes analisar a reputação da entidade que se deseja contratar e vice-versa. Desta forma, o sistema de reputação permite entidades (clientes) estabelecerem relações de confiança com outras entidades (provedores de serviços) e vice-versa.

O sistema de reputação deverá refletir a trajetória das entidades, permitindo diferenciar o comportamento de cada uma conforme o grau de satisfação ou qualidade de experiência (QoE - *Quality of Experience*[8]) informada pelas

[8] **Qualidade da Experiência** (QoE) é uma medida subjetiva. Mede o desempenho dos serviços a partir da perspectiva dos utilizadores [RAHRER, et. al., 2006].

entidades envolvidas e a qualidade de serviço (QoS – *Quality of Service*[9]) mensurada de alguma forma por alguma entidade da rede a ser abordada em trabalhos futuros.

Tanto as entidades que atuam como clientes como as provedoras de serviços deverão atribuir uma qualificação à negociação, informando se o contrato estabelecido foi bem sucedido ou não. Ou seja, ao finalizar um contrato, cada entidade deverá qualificar a sua contraparte informando um valor numérico de "0" a "10", que irá quantificar a avaliação subjetiva (QoE), devendo a avaliação objetiva (QoS) ser informada pela entidade responsável por gerenciá-la. Além da avaliação numérica, as entidades envolvidas poderão adicionar observações sobre a contratação ou prestação de serviços, se assim desejarem.

Perceba que os serviços contratados que melhor atenderem a demanda e honrarem o contrato estabelecido receberão melhores notas e consequentemente, terão maior chance de serem novamente selecionados e contratados. Desta forma, cria-se um mecanismo de evolução no ambiente digital. No entanto, para que o modelo de reputação funcione, as entidades ou serviços precisam manter o mesmo identificador durante todo o período de "vida". Caso uma entidade ou serviço "morra" e retorne com outro identificador, suas informações anteriores poderão não ser utilizadas [PELLISSARI, et. al., 2005].

5.1.3 CDS – Compilation and Decompilation System

De uma forma bem simples, um compilador é um programa que recebe como entrada um texto em alguma linguagem de programação (código fonte) e o traduz para um programa equivalente em outra linguagem (linguagem objeto) [AHO, et. al., 2008]. O processo inverso é realizado pelo descompilador [CIFUENTES, et. al., 1994]. A **figura 5.3** ilustra o processo realizado por esse sistema.

[9] **Qualidade de Serviço** (QoS) é uma medida objetiva. Mede o desempenho a nível de pacotes a partir da perspectiva da rede [RAHRER, et. al., 2006].

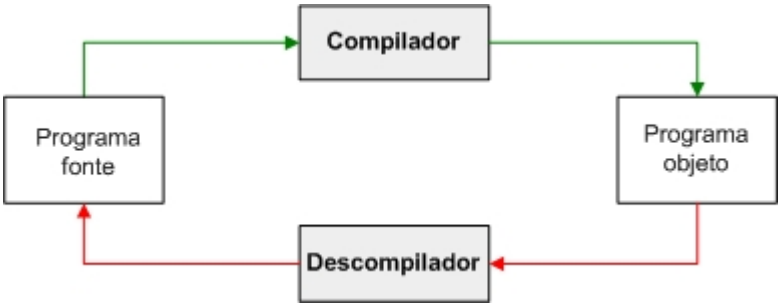

Figura 5.3: *Sistema de compilação e descompilação.*

O módulo de compilação é responsável por compilar códigos fonte (entidades no estado de informação) criando serviços virtuais. Ou seja, o compilador é o responsável por gerar uma "semente" de um serviço, que se executado "ganha vida" (entidades no estado de máquina). Os códigos fonte poderão ser criados manualmente pelo usuário ou de forma autonômica por entidades virtuais, sendo possível inclusive criar um código que coordene a invocação a serviços contratados, idéia similar à programação modular[10].

Nesta fase, a entidade OBS desempenha uma função primordial, visto que os serviços no estado de informação ainda não estão instanciados, sendo impossível que estes representem seus interesses e negociem sua contratação. Diante desta situação, o OBS será responsável por negociar a contratação do serviço, estabelecer o SLA junto a entidades contratantes, bem como iniciar o processo de compilação do serviço junto ao CDS. Uma vez compilado e instanciado, este poderá por si só realizar negociações futuras, bem como cumprir todos os contratos estabelecidos pelo OBS em seu nome.

O módulo de descompilação é responsável por realizar o caminho reverso, traduzindo o código executável em código fonte. Propõe-se que este processo seja realizado somente pelos proprietários dos serviços. Isto é, assim como nos dias atuais é vedado ao licenciado fazer ou permitir engenharia reversa[11] ou descompilação de programas, exceto quando requerido por lei para

[10] **Programação modular** é um método de design de software que divide um programa em componentes ou módulos.Os módulos podem ser desenvolvidos de forma independente, testados e refinados [WISEGEEK, 2011].

[11] **Engenharia reversa** visa analisar o software com o objetivo de recuperar seu projeto e sua especificação. O código fonte do software geralmente está disponível para que seja realizado o processo de engenharia reversa. Contudo, se até o código fonte foi perdido, a engenharia reversa irá começar com o código executável [SOMMERVILLE, 2003].

interoperacionalidade [JABUR, 2007], o contratante não poderá descompilar um serviço que não seja de sua autoria.

Perceba que o descompilador preserva a essência do serviço na forma de informação (código fonte). Desta forma, ele pode, por exemplo, ser utilizado para produzir um serviço novo, com manutenção mais fácil.

5.2 SLA – Service Level Agreement

Um SLA é um documento padronizado que formaliza a contratação de um serviço negociado entre as partes (provedor e cliente). Este documento deve especificar os objetivos e requisitos mínimos aceitáveis para o serviço negociado de forma que o provedor de serviços se comprometa a atender.

A questão contratual se divide basicamente em duas etapas: (*i*) definição e assinatura do SLA, que tem por objetivo definir o escopo do serviço; responsabilidades do cliente e do provedor de serviços; serviços pressupostos e valores; e (*ii*) gerenciamento do SLA, que é responsável por monitorar e garantir que os objetivos do SLA sejam cumpridos, bem como definir ações (multas ou políticas de recompensa, por exemplo) a serem tomadas caso isso não ocorra.

Para melhor entendimento, a **figura 5.4** redesenhada de [ANDRIEUX, et. al., 2007] ilustra a estrutura de um SLA para *WEB Services*.

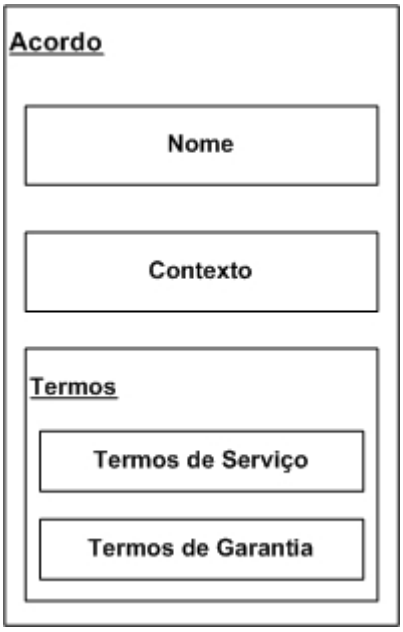

Figura 5.4: *Estrutura do SLA* ***[ANDRIEUX, et. al., 2007].***

Na **figura 5.4** existem basicamente três blocos (Nome, Contexto e Termos). O primeiro refere-se ao nome do SLA (Nome). O segundo contém os dados que compõe o acordo, como nome dos participantes e duração do contrato (Contexto). O terceiro referente aos termos (Termos), expressa o consenso e as obrigações definidas entre as partes, sendo dividido em dois tipos: Termos de Serviço, que descreve as informações necessárias para identificar o serviço [ANDRIEUX, et. al., 2007], por exemplo, se o provedor de serviço possui mais de uma funcionalidade (criptografia e autenticação). Termos de garantia, que especifica a qualidade de serviço acordada entre as partes, tais como tempo de recuperação de falha ou tempo máximo de indisponibilidade do serviço. Sistemas de gerenciamento podem utilizar os termos de garantia para monitorar o serviço e fazer cumprir o acordo [ANDRIEUX, et. al., 2007].

Neste trabalho, para o estabelecimento do SLA, usa-se a forma de negociação de rodada única, sem contra-ofertas. Ou seja, uma parte faz uma oferta de acordo, e a outra parte simplesmente aceita ou rejeita. Ressalta-se ainda que não haja restrições para que apenas uma das partes realize ofertas de SLAs. Isto é, em certos casos, será conveniente que o cliente defina os termos do acordo, e em outros o provedor do serviço. Situação ilustrada por meio da **figura 5.5** onde o provedor realiza a oferta do SLA e na **figura 5.6** onde o SLA é ofertado pelo cliente.

Por fim, para que o SLA seja estabelecido, uma das partes deverá publicar seu identificador no PSS, que naturalmente irá publicá-lo no GIRS, devendo sua contraparte assiná-lo. Posteriormente, será necessário medir o serviço sistematicamente, considerando os aspectos de qualidade, tecnologia e negócio, devendo a gerência do SLA ser realizada de alguma forma a ser abordada em trabalhos futuros.

5.3 Mapeamentos para a Arquitetura

Nesta seção, utiliza-se a flexibilidade de mapeamentos GIRS para mostrar as relações ilustradas nas **figuras 5.5** e **5.6**. Os mapeamentos são implementados através de registros em *hash* distribuídos. A **tabela 2** contém exemplos de alguns mapeamentos para a arquitetura proposta com as suas chaves de entrada, valores de saída, e o tempo de vida destes mapeamentos (TTL).

Tabela 2: *Exemplos de mapeamentos para a arquitetura que podem ser acomodados no GIRS.*

Mapeamento	**Chave**	**Valor (XML)**		
		Tipo	**Descrição**	**TTL**
<ID-Serviço; ID-Descritor>	ID-Serviço	1	IDs de descritores associados a um serviço.	1
<ID-Descritor; ID-Serviço>	ID-Descritor	2	IDs de serviços associados a um dado descritor.	1
<ID-Serviço; ID-Nome>	ID-Serviço	3	Relação de identificadores de nomes associados a um único serviço.	1
<ID-Nome; ID-Serviço>	ID-Nome	4	Relação de serviços que possuem exatamente o mesmo nome.	1
<ID-Nome; ID-Descritor>	ID-Nome	5	Relação dos descritores de objetos de serviços associados ao ID-Nome.	1
<ID-Descritor; ID-Nome>	ID-Descritor	6	Relação de nomes associados ao ID-Descritor como “Criptografia” e “Autenticação compartilhada”, por exemplo.	1
<ID-Serviço; ID-Substrato>	ID-Serviço	7	Relação de identificadores de substrato onde se encontra executando ou armazenado o serviço requisitado.	1
<ID-Substrato; ID-Serviço>	ID-Substrato	8	Relação de serviços encontrados em um determinado substrato.	1
<ID-Descritor; ID-Substrato>	ID-Descritor	9	Relação de identificadores de substrato que possuem o descritor de um serviço.	1
<ID-Substrato; ID-Descritor>	ID-Substrato	10	Relação de descritores de serviços disponíveis no substrato identificado pelo ID-Substrato.	1
<ID-SLA; ID-Serviço>	ID-SLA	11	Relação dos identificadores de serviços que fazem parte do SLA com este ID.	1
<ID-Reputação; ID-Entidade>	ID-Reputação	12	Relação dos identificadores de entidades associados à reputação com este ID.	1
<ID-Entidade; ID-Reputação>	ID-Entidade	13	Relação dos identificadores de reputação associados a uma determinada entidade.	1
<ID-CO; ID-Nome>	ID-CO	14	IDs de nomes associados a uma única CO.	1
<ID-Nome; ID-CO>	ID-Nome	15	Relação de COs que possuem exatamente o mesmo nome.	1
<ID-CO; ID-Descritor>	ID-CO	16	Relação de descritores de serviços associados a uma CO.	1
<ID-Descritor; ID-CO>	ID-Descritor	17	Relação de COs associadas a um dado descritor de serviço.	1
<ID-CO; ID-Entidade>	ID-CO	18	Relação de entidades que assinaram ou publicaram uma dada CO.	1
<ID-Entidade; ID-CO>	ID-Serviço	19	Relação de COs que uma dada entidade assinou ou publicou.	1
<ID-CO; ID-Substrato>	ID-CO	20	Relação de Substratos que possuem o conteúdo de uma CO.	1
<ID-Substrato; ID-CO>	ID-Substrato	21	Relação COs armazenadas em um determinado substrato.	1
<ID-Substrato; ID-Nome>	ID-Substrato	22	Relação de nomes de serviços disponíveis em um dado substrato.	1

A **tabela 2** relaciona 22 tipos de mapeamentos que representam as ligações dinâmicas entre os identificadores de Serviços, Descritores, Nomes, SLA, CO, Substrato, Entidade e Reputação, como ilustrado nas **figuras 5.5** e **5.6**.

Para exemplificar, o mapeamento 1 é utilizado para se obter o identificador do descritor (ID-Descritor) relacionado a um dado serviço (ID-Serviço). No mapeamento 2 deseja-se obter uma lista de serviços associados a um dado descritor, assim entra-se com a chave ID-Descritor e se obtém, por exemplo, a lista (ID-Serviço 1,..., ID-Serviço 6). O mapeamento 21 é utilizado para obter a lista de Oportunidades de Contrato armazenadas em um substrato, como por exemplo, "Edital Criptografia.xml" e "Edital Corretor Ortográfico.xml".

5.4 Localização e Contratação de Serviços em um e em Vários Domínios

Nesta seção apresentam-se três estudos de caso genéricos utilizando a arquitetura proposta para delinear o processo de localização e contratação de serviços em um ou mais domínios. Neste contexto, todo o relacionamento entre as entidades é baseado no Sistema Publica Assina (PSS), no qual as entidades requerentes publicam a posse do serviço ou conteúdo e requisitam valores associados às chaves através de assinaturas de serviços ou conteúdos. Não será considerada a situação de falha no serviço devido ao fato desse assunto não fazer parte dos objetivos da proposta.

É importante ressaltar que antes de todo o processo de localização e contratação de serviços, deve-se levar em conta questões de segurança e confiança entre as entidades, como por exemplo, pessoas, *hosts*, serviços, entre outras. Mas, para a diminuição da complexidade deste trabalho, abstrai-se que estas questões são sanadas por outros elementos da arquitetura, ficando, portanto, fora do escopo desta proposta. Ao término do contrato, as entidades envolvidas deverão avaliar sua contraparte e naturalmente publicar estas avaliações no sistema de reputação.

5.4.1 Localização e Contratação de Serviços em um Domínio

Considere o estudo de caso ilustrado pela **figura 5.5**, que representa o processo de localização e contratação de serviços em um domínio, no qual a entidade

(Cliente) está interessada em contratar um determinado serviço (*Criptografia.exe*) oferecido pela entidade (Provedor) e armazenado em algum nó virtual da rede. Neste ambiente, por padrão, todo serviço a ser publicado passa por uma série de interações responsáveis por identificá-lo, descrevê-lo e nomeá-lo. Para diminuir a complexidade deste exemplo, estas interações não estão presentes nesta figura.

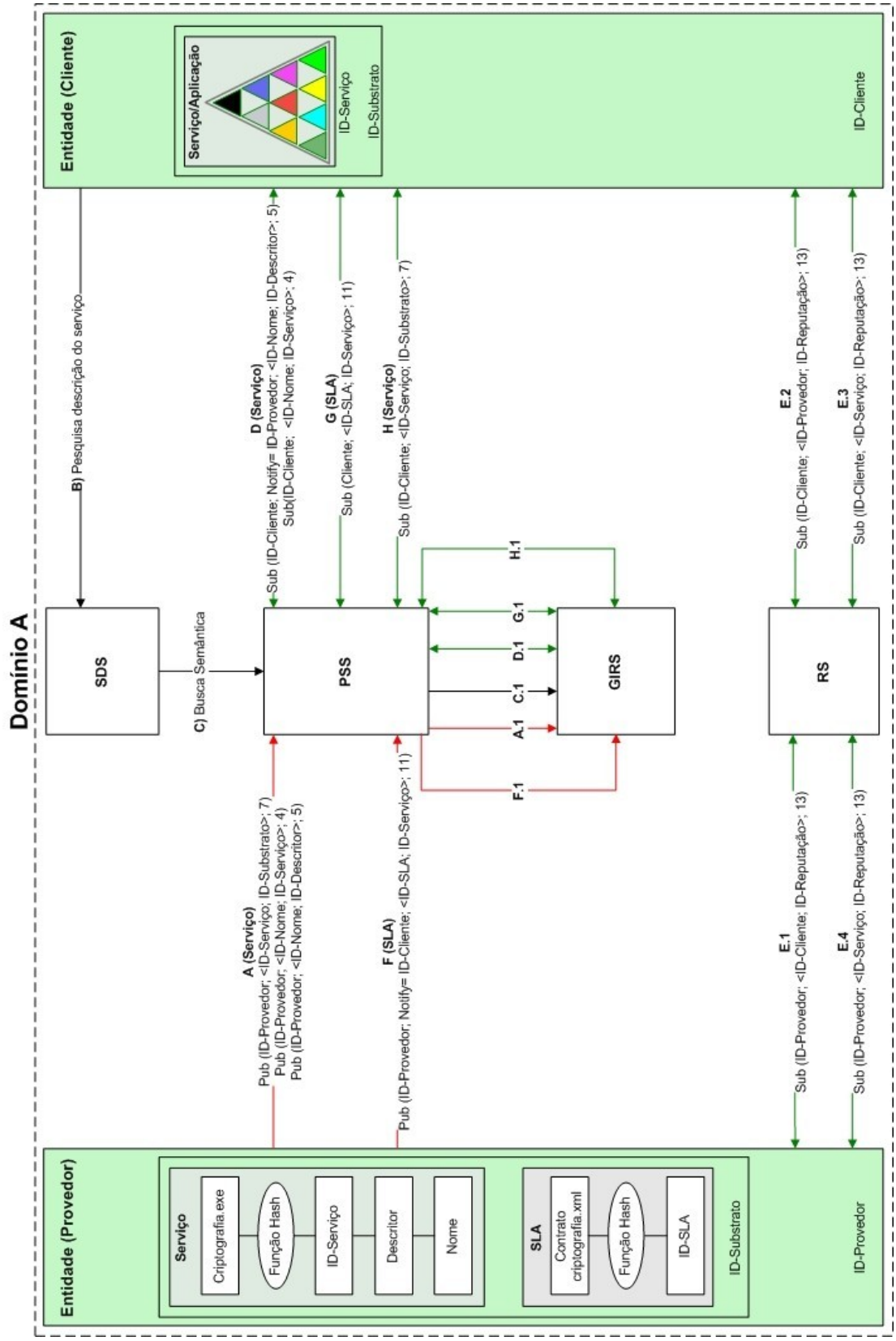

Figura 5.5: *Localização e contratação de serviço em um domínio.*

Após estas interações, a entidade responsável pelo serviço (Provedor) deve publicar no PSS o serviço (*Criptografia.exe*) que oferece (mapeamento 7 da **tabela 2**), o nome do serviço (mapeamento 4 da **tabela 2**) e seu descritor (mapeamento 5 da **tabela 2**), mapeamentos representados pelo (passo A). O PSS por sua vez, efetivará tais publicações no GIRS (passo A.1).

Dado que o serviço interessado já está publicado, a entidade requerente (Cliente) deve formular nomes adequados semanticamente (passo B) e por meio do SDS realizar uma busca semântica por serviços de seu interesse (passo C). Se houver resultados disponíveis, estes são retornados para o Cliente na forma de descritores e/ou nomes legíveis. Baseado nestas informações, o Cliente realiza uma análise criteriosa e assina um resultado (descrição e nome) cuja descrição mais se aproxime do serviço que deseja contratar, bem como notifica o Provedor sobre seu interesse em assinar determinado serviço (mapeamentos 5 e 4 da **tabela 2**, representados pelo passo D).

Após ser notificado, o Provedor por meio do RS verifica a reputação do Cliente e vice-versa (mapeamento 13 da **tabela 2**, representado pelos passos E.1 e E.2), bem como o cliente verifica a reputação do serviço que deseja contratar (mapeamento 13 da **tabela 2**, representado pelo passo E.3. Posteriormente é iniciado o processo de negociação, onde o cliente informa o identificador da aplicação (cluster de serviços representado pelo triângulo da cor preta) que irá utilizar o serviço que deseja contratar. A reputação da aplicação é então, consultada pelo provedor (mapeamento 13 da **tabela 2**, representado pelo passo E.4). São realizadas inúmeras interações até que haja um consenso entre as entidades envolvidas (etapa não ilustrada na **figura 5.5**).

A contratação do serviço é formalizada por meio de um SLA (*Contrato Criptografia.xml*). Isto é, o Provedor por meio do PSS publica no GIRS o identificador do SLA associado ao identificador do serviço que está sendo contratado e notifica o Cliente sobre sua publicação (mapeamento 11 da **tabela 2**, representado pelo passo F), possibilitando a este assinar o identificador do SLA (mapeamento 11 da **tabela 2**, representado pelo passo G).

Por fim, o identificador do resultado selecionado e o identificador do SLA são inseridos como parâmetros para a assinatura do serviço no GIRS. O serviço é disponibilizado ao Cliente, sendo realizado o mapeamento entre o identificador do serviço e o identificador do substrato responsável por seu armazenamento (mapeamento 7 da **tabela 2**, representado pelo passo H).

5.4.2 Localização e Contratação de Serviços em um Domínio – Publicação de COs

Considere o estudo de caso ilustrado pela **figura 5.6**, que representa o processo de localização e contratação de serviços em um domínio, no qual a entidade (Cliente) está interessada em contratar um determinado serviço (*Criptografia.exe*), mas não encontrou um serviço condizente com sua pesquisa. Desta forma, é necessário publicar uma CO (*Edital Criptografia.xml*) com o intuito de despertar o interesse de alguma entidade em criar o serviço conforme requisitos solicitados.

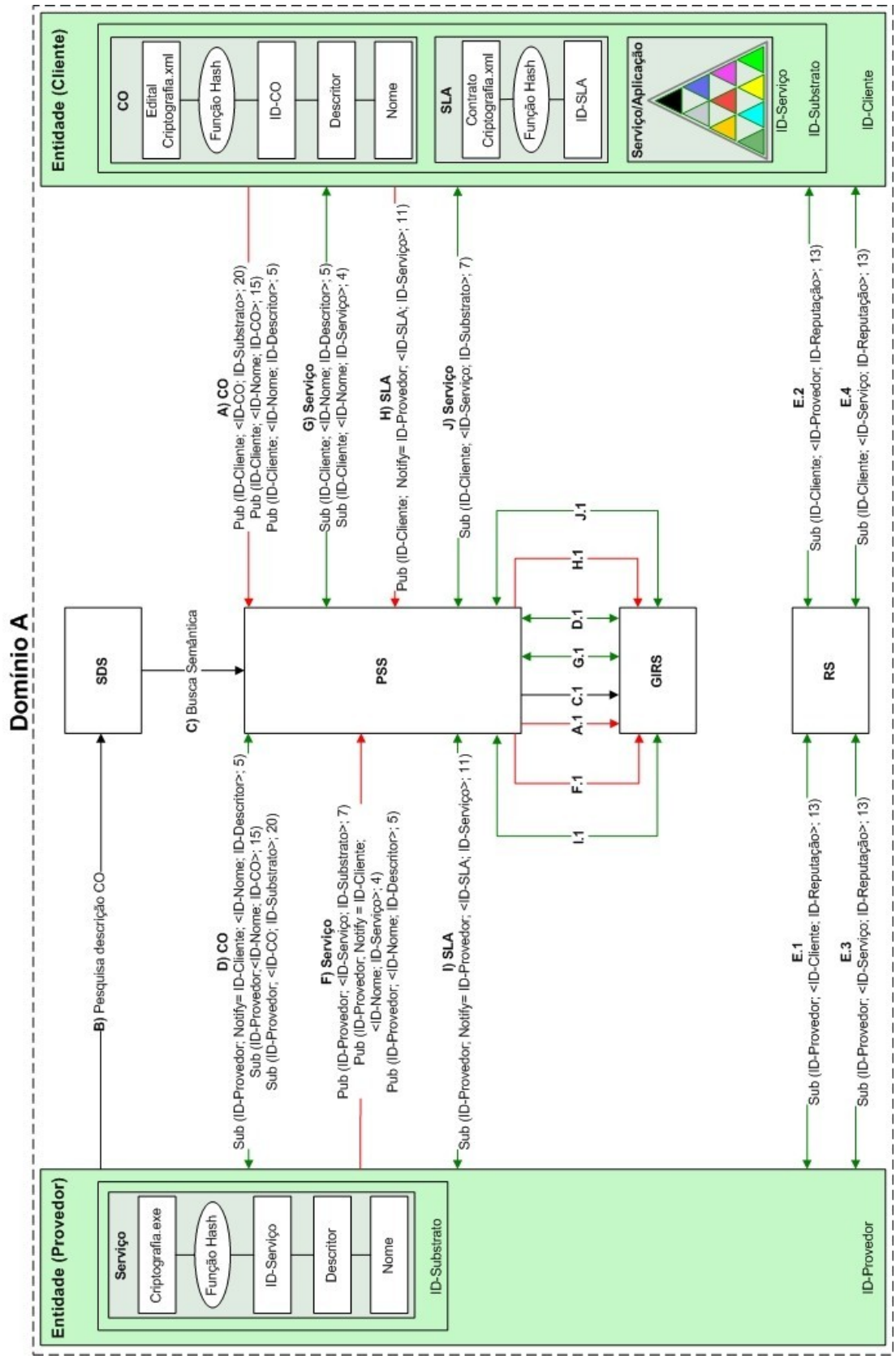

Figura 5.6: *Publicação de CO e contratação de serviço em um domínio.*

O Cliente responsável pela Oportunidade de Contrato (*Edital Criptografia.xml*) deve publicar no PSS o conteúdo da CO (mapeamento 20 da **tabela 2**), o nome da CO (mapeamento 15 da **tabela 2**) e seu descritor (mapeamento 5 da **tabela 2**), (passo A). O PSS por sua vez, efetivará tais publicações no GIRS (passo A.1).

Com a CO já publicada, o Provedor deve formular nomes adequados semanticamente (passo B) e por meio do SDS realizar uma busca semântica por conteúdos de seu interesse (passo C). Se houver resultados disponíveis, estes são retornados para o Provedor na forma de descritores e/ou nomes legíveis. Baseado nestas informações, o Provedor realiza uma análise criteriosa e assina um resultado cuja descrição mais se aproxime do conteúdo que deseja acessar, bem como notifica o Cliente que deseja atender determinada CO (mapeamentos 5, 15 e 20 da **tabela 2**, representados pelo passo D).

Após ser notificado, o Cliente por meio do RS verifica a reputação do Provedor e vice-versa (mapeamento 13 da **tabela 2**, representado pelos passos E.1 e E.2. Posteriormente é iniciado o processo de negociação, onde o cliente informa o identificador da aplicação (*cluster* de serviços, triângulo composto por triângulos de várias cores, onde cada triângulo representa um serviço e o triângulo de cor preta representa a junção de todos eles) que irá utilizar o serviço que deseja contratar e o Provedor informa o identificador do serviço que será utilizado, caso este já exista. A reputação da aplicação é então, consultada pelo Provedor e a reputação do serviço consultada pelo Cliente (mapeamento 13 da **tabela 2**, representado pelos passos E.3 e E.4). São realizadas inúmeras interações até que haja um consenso entre as entidades envolvidas (etapa não ilustrada na **figura 5.6**).

Caso o serviço solicitado ainda não exista, o Provedor poderá desenvolver, compilar, identificar e publicar o novo serviço (*Criptografia.exe*) no PSS (mapeamento 7 da **tabela 2**), o nome do serviço, bem como notifica o cliente da publicação (mapeamento 4 da **tabela 2**), e por fim, publicar o descritor do serviço (mapeamento 5 da **tabela 2**), (passo F). O PSS por sua vez, efetivará tais publicações no GIRS (passo F.1).

Dado que a publicação foi realizada com sucesso, o Cliente assina o descritor e nome do serviço publicado (mapeamento 5 e 4 da **tabela 2**, representado pelo passo G) e formaliza a contratação do serviço por meio de um SLA (*Contrato Criptografia.xml*). Isto é, o Cliente por meio do PSS publica o identificador do SLA no GIRS e notifica o Provedor sobre sua publicação (mapeamento 11 da **tabela 2**, representado pelo passo H). Então, o Provedor assina o identificador do SLA e notifica o cliente sobre sua assinatura (mapeamento 11 da **tabela 2**, representado pelo passo I).

Uma vez que os identificadores referentes ao nome, descritor do serviço e SLA foram assinados, estes são inseridos como parâmetros para a assinatura do serviço no GIRS. Então, o serviço é disponibilizado ao Cliente, sendo realizado o mapeamento entre o identificador do serviço e o identificador do substrato responsável por seu armazenamento (mapeamento 7 da **tabela 2**, representado pelo passo J).

Observe que o processo de publicação e assinatura de conteúdos e serviços é similar, podendo os sistemas ilustrados nas **figuras 5.5** e **5.6** serem utilizados tanto para conteúdo quanto para serviços. Outra observação importante, é que ambas as entidades (Cliente ou Provedor) podem publicar SLAs, permitindo desta forma a flexibilidade da proposta.

5.4.3 Localização e Contratação de Serviços em Vários Domínios

Diferentemente dos cenários apresentados até o momento, a **figura 5.7** ilustra a contratação de serviços por meio do SDS e publicação de COs em diversos domínios.

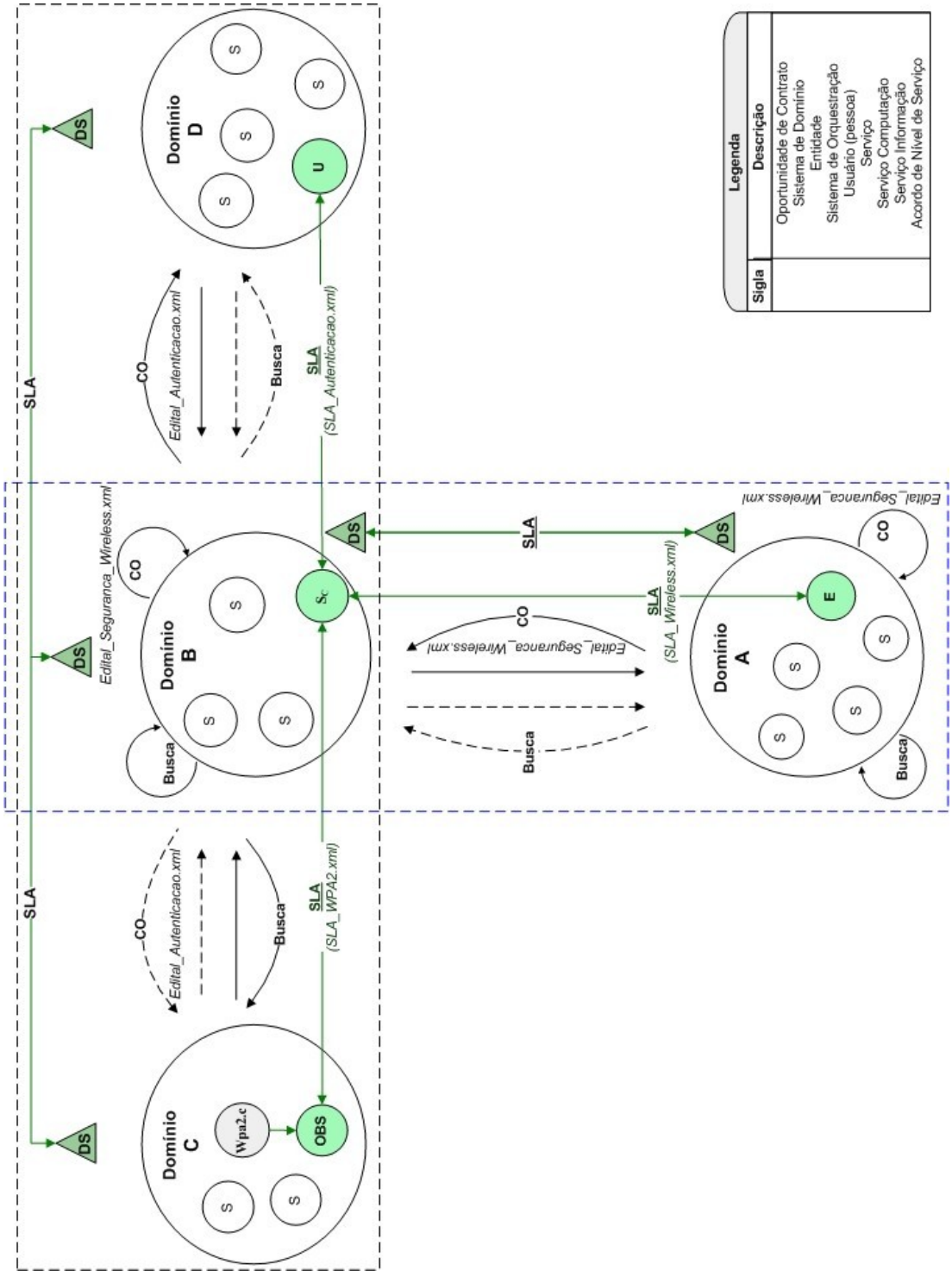

Figura 5.7: *Publicação de CO e contratação de serviço em múltiplos domínios.*

Na **figura 5.7** o tracejado da cor azul limita a parceria entre os domínios (A) e (B). Já o tracejado da cor preta limita a parceria do domínio (B) com os domínios (C) e (D). As parcerias ou cooperações entre domínios são resultados dos SLAs estabelecidos entre seus respectivos Sistemas de Domínio (DS).

Neste cenário, a entidade E deseja contratar um serviço específico (*Segurança Wireless*). Ela realiza uma busca semântica em seu domínio (A) e no domínio parceiro (B). No entanto, não foi retornado nenhum resultado condizente com a pesquisa. A alternativa encontrada pela entidade foi publicar uma CO (*Edital_Segurança_Wireless.xml*) em ambos os domínios. Esta CO descreve basicamente o interesse da entidade E em contratar um serviço de segurança para redes *wireless* (*Segurança Wireless*), que seja composto por um serviço de autenticação compartilhada e um serviço de criptografia (*WPA2*).

No domínio (B), um serviço capaz de negociar (Serviço Computação - S_C) realiza uma busca por oportunidades de contrato em aberto e como resultado obtém o nome e/ou descritor referente a CO publicada pela entidade E. Em seguida, assina o conteúdo da CO e verifica as características e condições para contratação do serviço desejado. Mesmo não havendo serviços condizentes no domínio (B), a entidade S_C é capaz de realizar sua composição, desde que tenha os serviços necessários à disposição. Então, realiza uma busca junto aos domínios (C) e (D), encontrando no domínio (C) o serviço (*WPA2.c*) condizente com a busca, mas no estado de informação, sendo seus interesses representados por uma entidade OBS. É realizado o processo de negociação e estabelecido um contrato (*SLA_WPA2.xml*) entre as entidades OBS e S_C, sendo o serviço compilado e disponibilizado à entidade S_C.

No domínio (D), não foi encontrado nenhum resultado condizente com a pesquisa. Desta forma, a entidade S_C publica uma CO (*Edital_Autenticacao.xml*) nos domínios (C) e (D). No domínio D, a entidade (Usuário - U), que é um ser humano, realiza uma busca por Oportunidades de Contrato e obtém como resultado a CO publicada pela entidade S_C. Posteriormente, assina o conteúdo e verifica as características e condições que o serviço de autenticação deverá atender. Em seguida desenvolve manualmente o serviço de autenticação compartilhada, realiza sua identificação, nomeação, descrição e publicação. Em seguida, são realizados os processos de negociação e contratação (*SLA_Autenticacao.xml*) entre as entidades U e S_C, bem como, o fornecimento do serviço requerido.

Com os serviços à disposição, a entidade S_C cria um *script* de serviços, que quando compilado, dará origem a um *cluster* de serviços que poderá atender a CO publicada pela entidade E. O *cluster* de serviços será identificado, publicado, negociado e estabelecido um SLA (*SLA_Wireless.xml*) entre as entidades E e S_C. Estando estas etapas concluídas, o serviço será fornecido à entidade E.

Observe que, quanto mais parceiros o domínio possuir, mais apto a compor serviços semanticamente suas entidades serão. Além do mais, a proposta trata de uma arquitetura autossimilar, que independe de escala, pois a mesma solução pode ser usada em níveis diferentes da hierarquia de domínios.

5.5 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* do ponto de vista estrutural (estático) e comportamental (dinâmico) [PENDER, 2004]. Nas seções seguintes serão apresentados alguns diagramas UML para a especificação de *software* do OBS e GS.

5.5.1 Diagramas de Caso 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. Alguns casos de uso podem interagir com outros casos de uso por meio de um relacionamento modelado por setas de dependência.

Serão apresentados nas subseções seguintes, sete cenários que evoluem em complexidade e abordam como é feita a publicação e assinatura de COs, bem como a contratação de serviços.

5.5.1.1 Publicação de CO

A **figura 5.8** ilustra a interação entre os atores OBS mediador do ator *Serviço_Usuário* (serviço incapaz de negociar a composição de um novo serviço) com os atores PSS e o GIRS que interagem para publicar a Oportunidade de Contrato (CO).

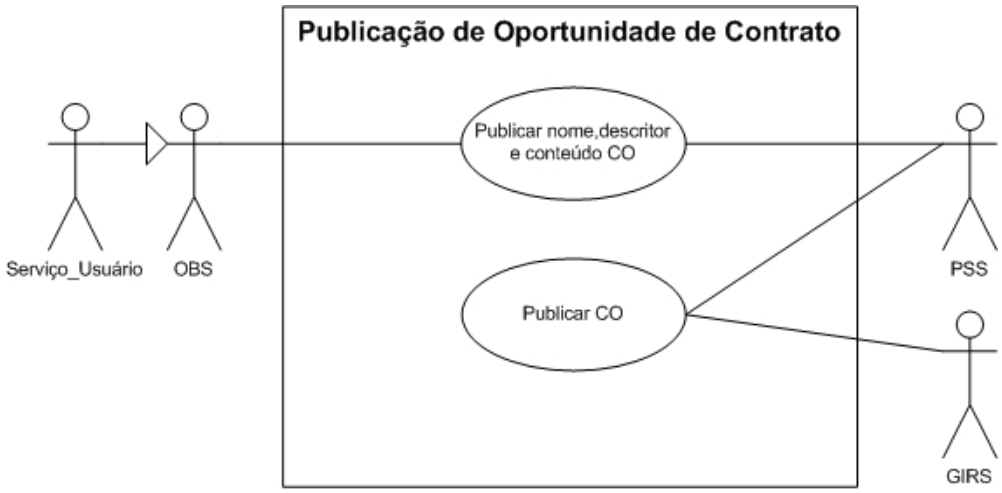

Figura 5.8: *Publicação da Oportunidade de Contrato pelo OBS.*

Ainda na **figura 5.8**, cada caso de uso representa uma interação entre os atores, onde o caso de uso:

- Publicar nome, descritor e conteúdo da CO: representa a publicação do nome, descrição e conteúdo da Oportunidade de Contrato no Sistema Publica Assina.

- Publicar CO: representa a publicação dos identificadores referentes ao nome, descritor e conteúdo da Oportunidade de Contrato feita pelo PSS no GIRS.

5.5.1.2 Busca e Assinatura de COs Publicadas

A **figura 5.9** ilustra a interação do ator OBS com os atores SDS, PSS e GIRS para pesquisar, assinar e mapear Oportunidades de Contrato de seu interesse.

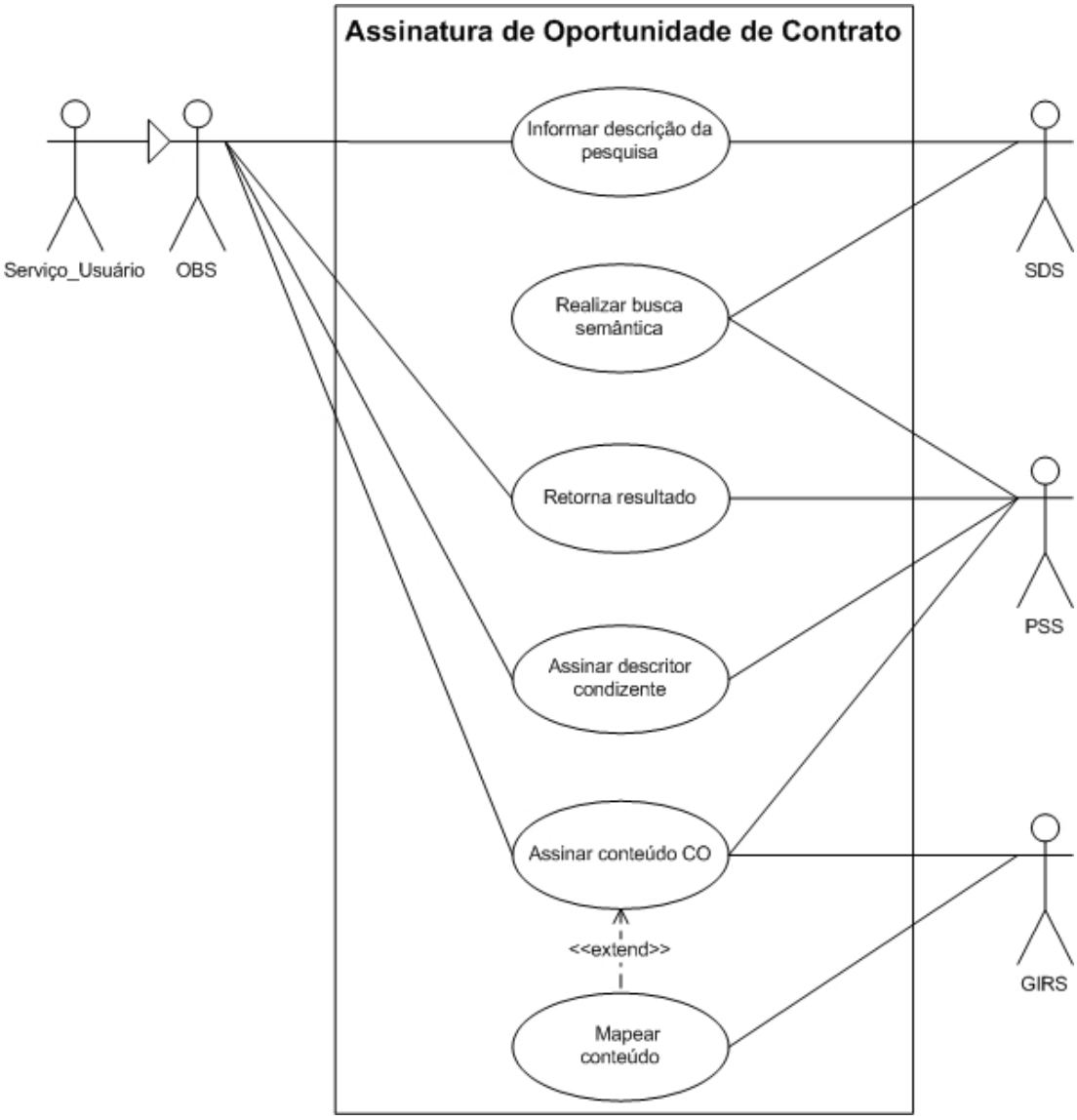

Figura 5.9: *Busca e assinatura da Oportunidade de Contrato pelo OBS.*

Ainda na **figura 5.9**, além dos casos de uso já definidos na **figura 5.8**, foram acrescentados os seguintes:

- Informar descrição da pesquisa: representa o interesse do ator OBS em uma determinada descrição de serviços ou conteúdos.

- Realizar busca semântica: representa a pesquisa e recuperação de entidades com base na semântica e descrição dos serviços ou conteúdos.

- Assinar descritor condizente: representa a assinatura do identificador do descritor do conteúdo ou serviço feita pelo ator *OBS* junto ao *PSS* após identificar resultados condizentes.

- Assinar conteúdo da CO: representa a assinatura do conteúdo da CO através de seu identificador, sendo este processo realizado pelo ator OBS junto ao PSS e por consequência, o GIRS.

- Mapear conteúdo: representa uma interação estendida do caso de uso *Assina Conteúdo/Serviço* desenvolvido pelo GIRS. Este caso de uso é responsável por mapear o identificador do conteúdo à entidade responsável por seu armazenamento. A interação deste caso de uso com o caso de uso *Assina Conteúdo/Serviço* é feita através de um relacionamento <<*extend*>>.

5.5.1.3 - Busca por Serviços e Publicação de CO

A **figura 5.10** ilustra a interação do ator OBS com os atores SDS, PSS e GIRS para pesquisar serviços/conteúdos e publicar Oportunidades de Contrato. Neste cenário, o ator OBS através do SDS realiza uma busca semântica por descritores de serviço, mas não encontra algum serviço condizente com sua pesquisa. Será necessário então, publicar uma CO junto ao PSS, que por sua vez, efetuará tal publicação no GIRS.

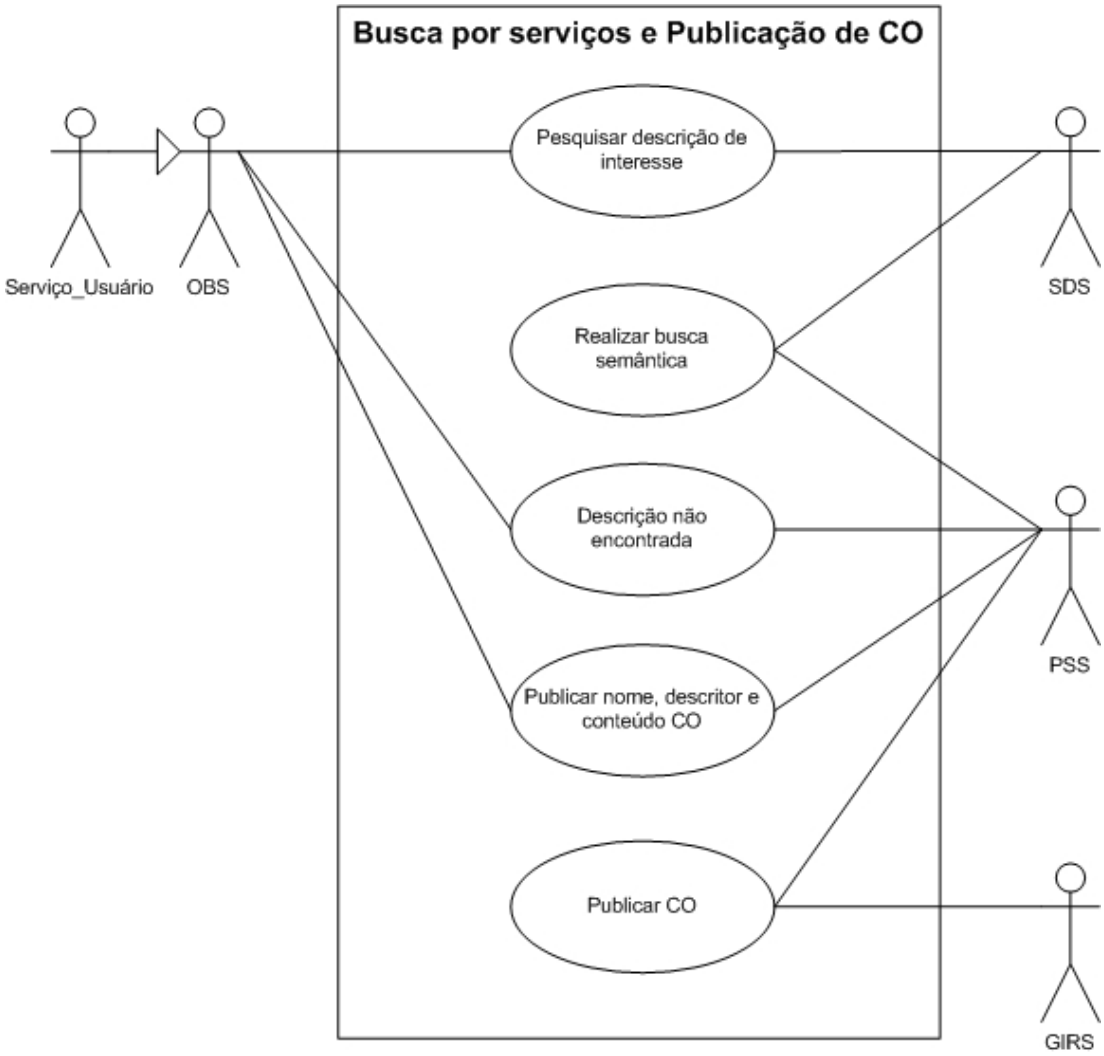

Figura 5.10: *Busca e publicação de Oportunidade de Contrato pelo OBS.*

Neste caso, além dos casos de uso já definidos nas **figuras 5.8** e **5.9**, foi acrescentado o caso de uso:

- Descrição não encontrada: significa que não foi encontrada nenhuma descrição de serviços/conteúdo condizente com a pesquisa realizada por meio do ator SDS junto ao PSS.

5.5.1.4: Busca e Assinatura de Serviços e/ou Conteúdos em Domínios Parceiros

Este cenário trata da busca que não retorna oportunidades em um domínio, e, portanto, avança para outro domínio. A **figura 5.11** ilustra a interação entre os atores Sistema de Domínio (DS –*Domain System*) que são os chefes dos domínios (A) e (B). No domínio (A), o ator OBS junto ao SDS e PSS realiza uma

busca semântica referente ao conteúdo ou serviço que deseja contratar, porém, nenhum resultado condizente é retornado. As alternativas para esta situação são: (*i*) publicar uma Oportunidade de Contrato no domínio (A) e/ou (*ii*) replicar a pesquisa para o domínio (B). Neste cenário, a pesquisa foi replicada no domínio (B), onde é então encontrado um serviço/conteúdo condizente com a CO em questão. A CO encontrada é assinada e mapeada através do OBS do domínio (A) junto ao PSS e GIRS do domínio (B).

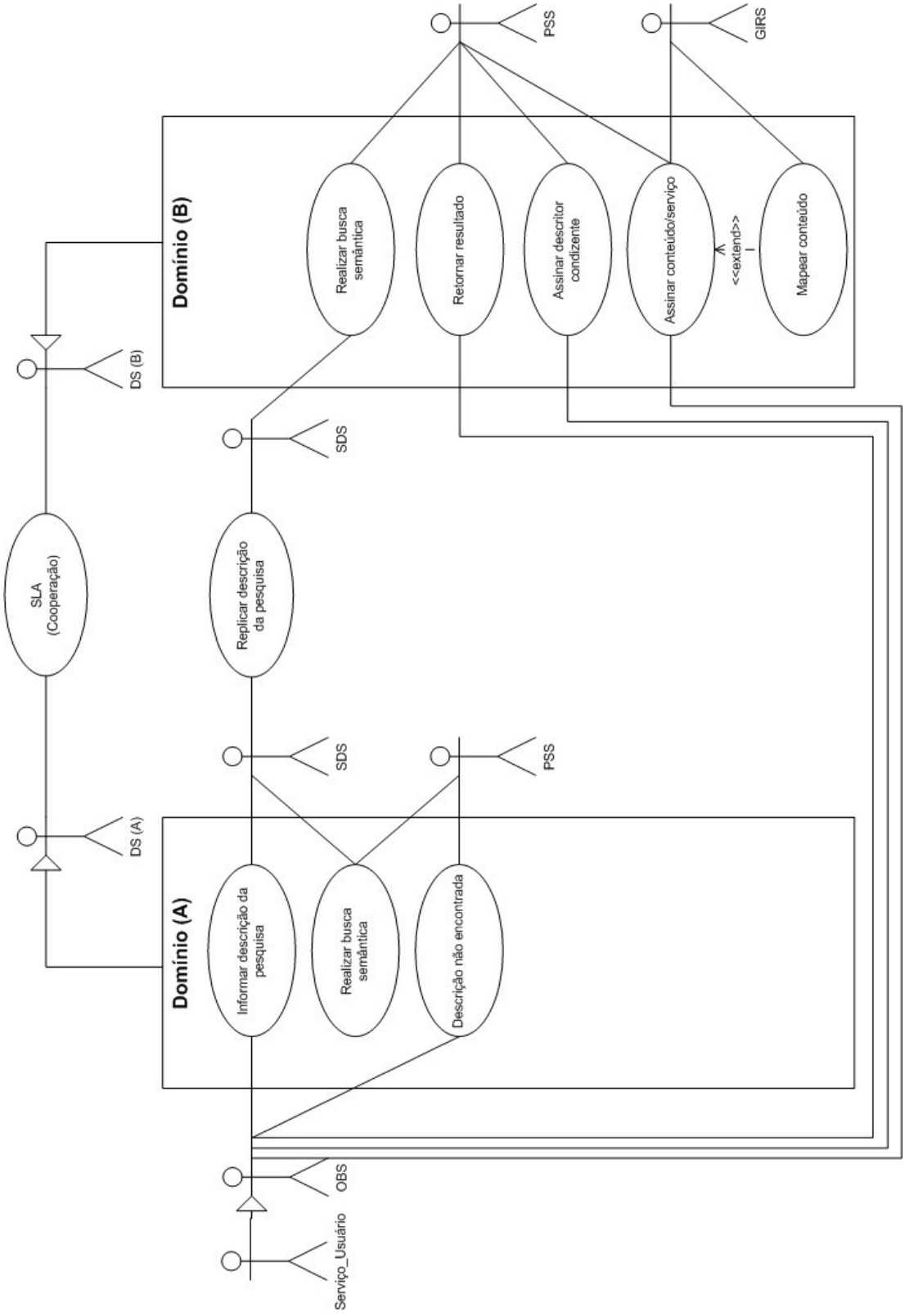

Figura 5.11: *Busca e assinatura de serviços e/ou conteúdos pelo OBS em domínios parceiros.*

Ainda na **figura 5.11**, além dos casos de uso já definidos nas **figuras 5.8** a **5.10**, foram acrescentados os seguintes:

- Replicar descrição da pesquisa: a descrição da pesquisa realizada no domínio (A) é replicada no domínio (B), sendo que os atores SDS de todos os domínios parceiros estão diretamente envolvidos.

- SLA (Cooperação): o Acordo de Nível de Serviço representa um contrato formal estabelecido entre os atores DS e define o nível de colaboração entre os domínios (A) e (B), que inclui buscas semânticas entre domínios.

5.5.1.5: Busca e Publicação de COs em Domínios Parceiros

A **figura 5.12** apresenta um cenário onde a descrição foi pesquisada em todos os domínios parceiros, mas nenhum deles dispunha de um serviço condizente em execução. A alternativa proposta é que a entidade OBS publique por um período determinado uma oportunidade de contrato junto ao PSS que irá replicá-la a todos os domínios parceiros.

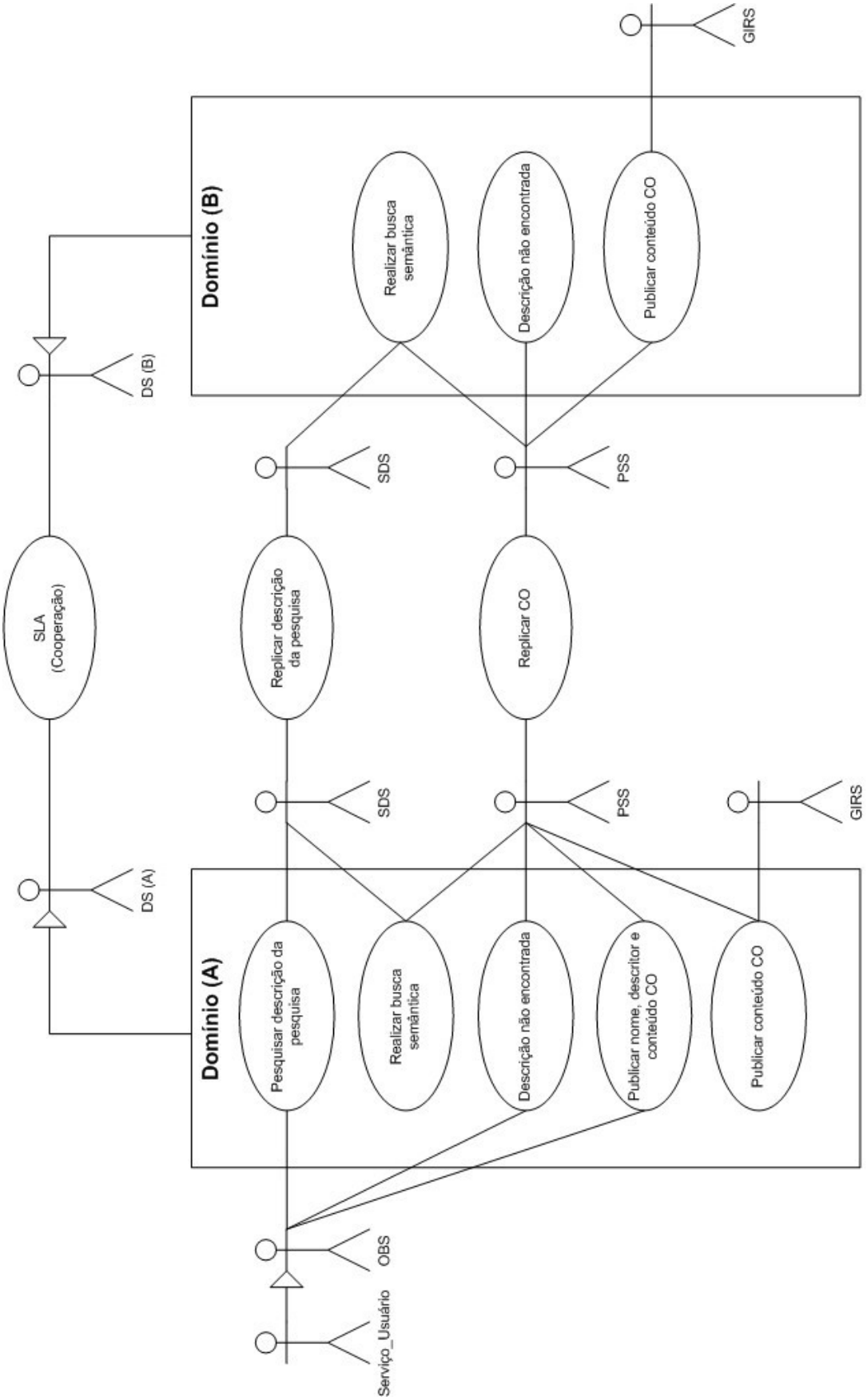

Figura 5.12: *Busca e publicação de Oportunidade de Contrato pelo OBS em domínios parceiros.*

Ainda na **figura 5.12**, além dos casos de uso já definidos nas **figuras 5.8** a **5.11**, foi acrescentado o seguinte:

- Replicar CO: o nome, descrição e conteúdo da Oportunidade de Contrato publicada no domínio (A) são replicados no domínio (B), sendo que os atores PSS de todos os domínios parceiros estão diretamente envolvidos.

5.5.1.6: Contratação de Serviços Publicados

Este cenário diz respeito à contratação de serviços publicados após a análise de oportunidades de contrato pelos serviços em questão. A **figura 5.13** ilustra a interação entre os atores OBS com o SDS, PSS e o RS.

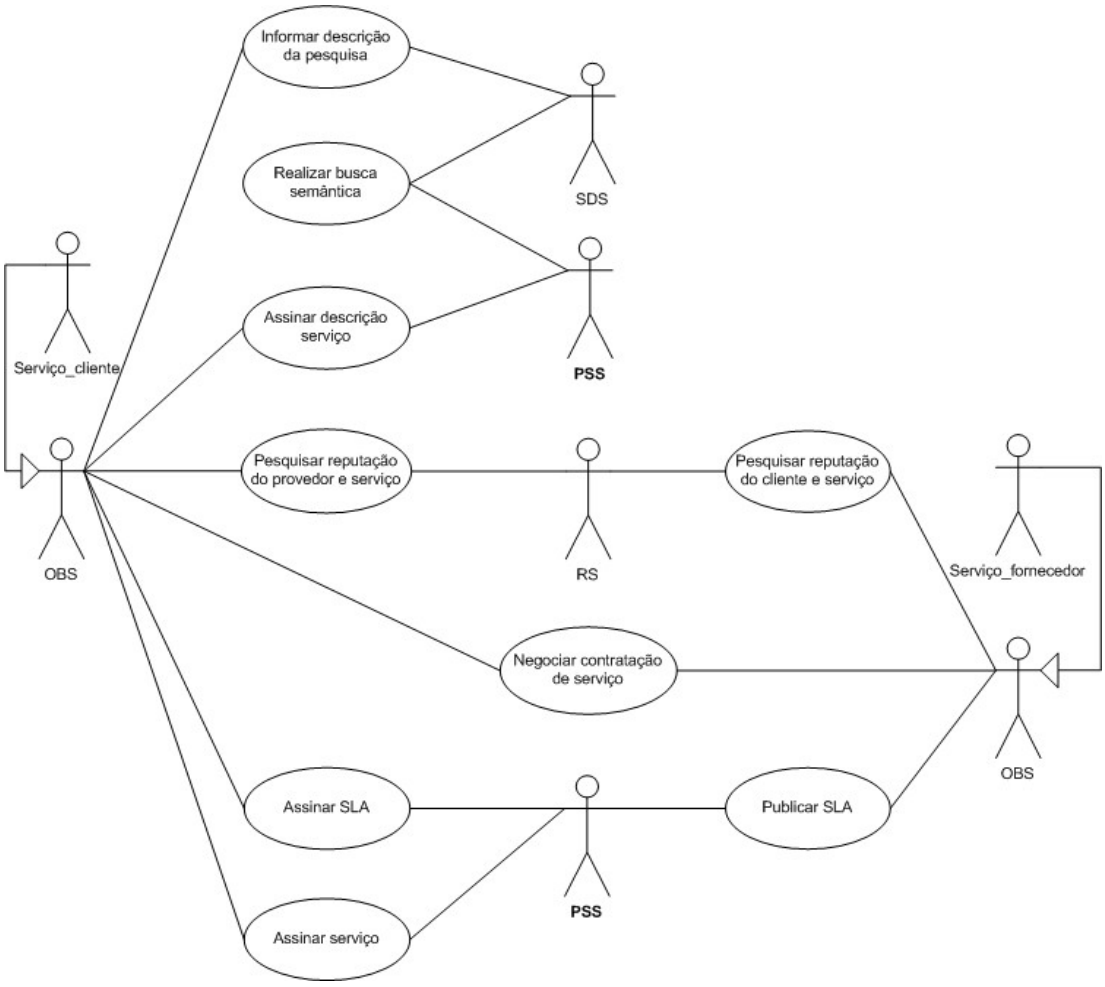

Figura 5.13: *Contratação de serviços publicados.*

Observe que tanto os atores *Serviço_cliente* e *Serviço_fornecedor* não possuem a capacidade de realizar negociações e formalizá-las, sendo necessária a

aparição de dois atores OBS para representá-los. Note que um dos serviços ou ambos podem ter a capacidade de negociar por conta própria os seus contratos. Neste caso, o OBS não precisa ser acionado tal qual acontece na **figura 5.13**. Após a negociação e contratação do serviço através de um SLA, este será mapeado e constantemente monitorado. Caso o serviço contratado se encontre no estado de informação (na forma de código fonte, *script* ou executável), ele deverá ser instanciado antes dos passos citados. Além dos casos de uso já definidos nas **figuras 5.8** a **5.12**, foram acrescentados os seguintes:

- Pesquisar reputação do provedor e serviço: possibilita pesquisar a reputação da entidade provedora de serviços, neste caso o OBS, bem como a reputação do serviço que esta representa, obtendo uma qualificação quantificada e justificada.

- Pesquisar reputação do cliente: possibilita pesquisar a reputação da entidade cliente, neste caso o OBS, bem como a reputação do serviço que esta representa, obtendo uma qualificação quantificada e justificada.

- Negociar contratação de serviço: por meio deste caso de uso as entidades envolvidas sondam informações umas das outras; buscam a aceitação de seus interesses e propostas; definem o acordo de prestação de serviços e como será feito seu monitoramento. O objetivo é obter um resultado consensual que satisfaça às expectativas de ambas as partes, estando estas convencidas de que argumentaram, foram entendidas e que estabeleceram um acordo satisfatório.

- Publicar SLA: Este caso de uso representa o Acordo de Nível de Serviço formalizado entre as entidades envolvidas e a publicação do seu identificador através do ator OBS (representante do *Serviço_fornecedor*) junto ao PSS.

- Assinar SLA: Este caso de uso representa a assinatura do identificador do Acordo de Nível de Serviço realizada por meio do ator OBS (representante do *Serviço_cliente*) junto ao PSS.

5.5.1.7: Contratação de Serviços não Publicados

A **figura 5.14** ilustra um cenário similar a **figura 5.13**, no entanto, com uma grande diferença. O ator OBS, representante do *Serviço_cliente* (aplicação *wireless*, por exemplo), deseja contratar um serviço que atenda as características e condições descritas na CO publicada por ele (Edital_Segurança_Wireless.xml, por exemplo). Já o ator OBS, representante dos atores *Serviço_fornecedor_1* (Criptografia.exe, por exemplo) e *Serviço_fornecedor_2* (Autenticação_Compartilhada.exe, por exemplo), ao assinar a CO demonstra seu interesse em compor e negociar o provimento do serviço requerido.

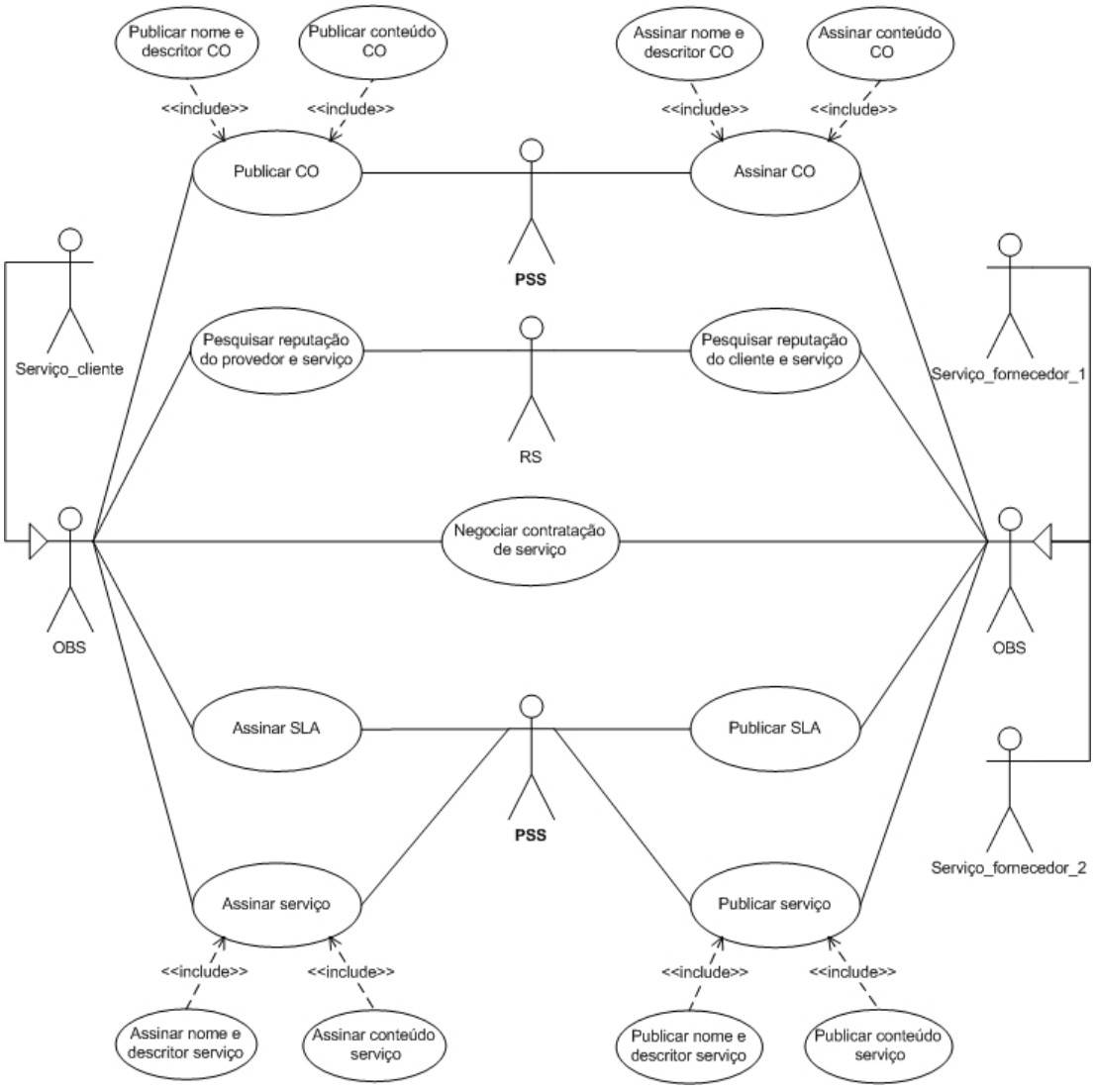

Figura 5.14: *Contratação de serviços não publicados.*

Observe que é possível que o ator OBS, representante dos serviços fornecedores, assine a CO publicada e componha o serviço especificado apenas com os serviços que ele representa, resultando assim, em um *cluster* de serviços que poderá vir a atender a demanda em aberto.

Dado que a CO foi assinada, as entidades OBS verificam a reputação umas das outras, bem como dos serviços que serão utilizados na composição do *cluster* de serviços e do serviço que irá utilizá-lo. Então, é iniciado o processo de negociação, sendo realizadas inúmeras interações até que haja um consenso entre as partes envolvidas. Por fim, a formalização da contratação do serviço ou *cluster* de serviços é realizada por meio de um SLA.

Conforme apresentado nos cenários 6 e 7, o processo de negociação pode ser inicializado de duas formas: (*i*) quando a entidade requerente deseja contratar um serviço já publicado ou (*ii*) quando a entidade provedora desenvolve um serviço para atender uma determinada Oportunidade de Contrato publicada pela entidade requerente. Concluída esta etapa, é estabelecido um Acordo de Nível de Serviço entre a entidade provedora de serviços (fornecedor) e a entidade requerente (cliente).

5.6 Ciclo de Vida

A palavra VIDA é definida no dicionário Aurélio, sob diferentes aspectos, nos quais o que mais está relacionado com o contexto deste trabalho é: "Vida é o conjunto de propriedades e qualidades graças às quais animais e plantas se mantêm em contínua atividade; existência". Relacionando esta definição com redes centradas em serviços, vida pode ser definida como a capacidade que os serviços possuem de se reproduzir, de se adaptar a novos ambientes e se organizar independentemente de agentes externos, como por exemplo, pessoas.

O modelo de ciclo de vida proposto para redes centradas em serviços possui fases que vão desde a criação até a finalização do serviço. A **figura 5.15** apresenta o serviço **S**, representado pelo círculo da cor preta no centro do labirinto e as quatro fases do seu ciclo de vida.

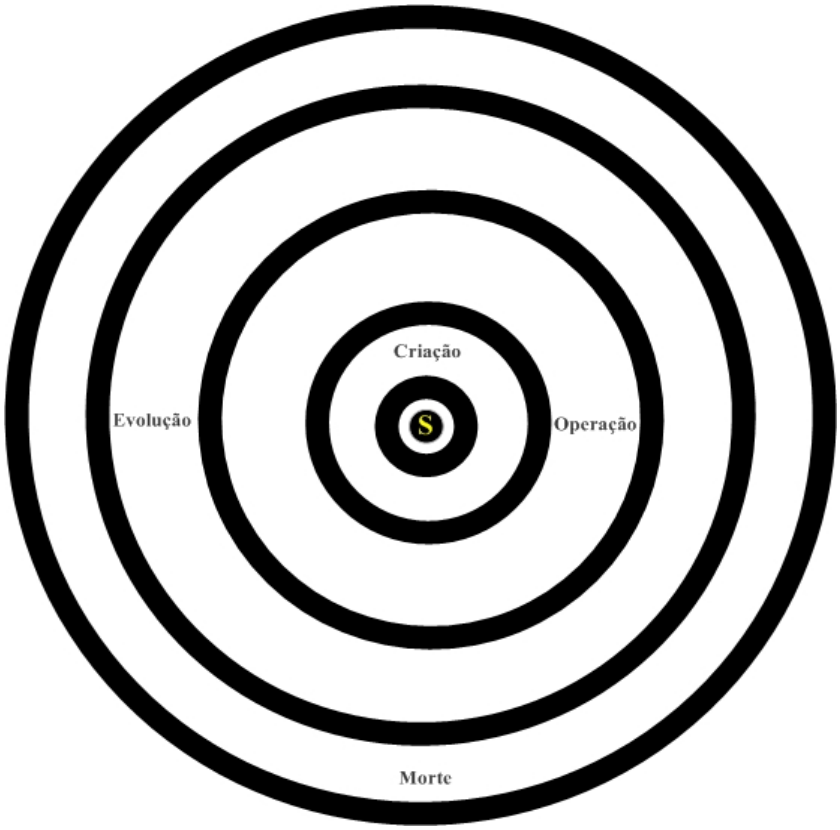

Figura 5.15: *Labirinto do ciclo de vida de serviços.*

- **Fase de criação:** é a fase inicial. Quando um serviço é criado, compilado, identificado e publicado.

- **Fase de operação:** é quando o serviço está realizando o processo para qual foi desenvolvido (computação de informações provenientes de outros serviços).

- **Fase de evolução:** durante a operação de serviços, evoluções podem ser necessárias, podendo adicionar e/ou substituir algum serviço. Etapas similares às especificadas na fase de criação poderão ser utilizadas nesta fase.

- **Fase de finalização:** existem duas causas que podem provocar a morte ou término do serviço. A primeira é quando ele já concluiu o propósito para qual foi criado e a segunda é quando o serviço compromete o domínio como um todo (segurança, por exemplo). Propõe-se que quando um serviço chegar ao fim da sua vida seja inicializado de alguma forma uma função **Destrutor**, que será responsável por realizar as tarefas que são necessárias executar quando

uma entidade deixa de existir, limpando todo o lixo eletrônico (desfazendo contratos, indireções, e até mesmo algumas publicações, entre outros).

5.7 Análise Comparativa

Esta seção apresenta as características mais relevantes do Sistema Genesis. A **tabela 3** resume as comparações entre a arquitetura proposta e os trabalhos estudados e comparados no Capítulo 3.

Tabela 3: *Comparação entre os Trabalhos Relacionados e a Arquitetura Proposta..*

Princípios	**CASCADAS**	**BIONETS**	**XIA**	**GS**
Descrição	GN abrange descrição semântica.	Semântica.	Nomes legíveis.	Nomes e descritores com alto grau semântico.
Identificação	Carece de identificação de serviços.	Carece de identificação de serviços.	Provê a identificação única das entidades de serviços, conteúdos e hosts.	Provê a identificação única das entidades de serviços e conteúdos.
Descoberta	Seleção de serviços usando GN/GA.	Não identificado.	Baseada na resolução de nomes para identificadores.	Baseado na estrutura de ontologias da rede e SDS.
Composição	Agregação/ diferenciação dinâmica, contextualizada.	Dinâmica, genótipo de serviços.	Dinâmica baseada em identificadores.	Diferentes formas de criar um serviço, tais como: manual e autonômica.
Localização transparente	Desacoplamento via publicação/ assinatura de eventos.	Não identificado.	Desacoplamento via servidor de *lookup*.	Usando o PSS e o DHTS.
Negociação	Direta usando GN/GA.	Não identificado.	Não possui.	Usando o PSS.
Contratação	Direta usando agentes DIET.	Não identificado.	Não possui.	SLA como parte integrante da arquitetura.
Confiança	Gestão de confiança e reputação.	Gestão de confiança e reputação.	Gestão de confiança.	Sistema de Reputação.
Mobilidade	Nativa.	Nativa.	Nativa.	Nativa.
Adaptabilidade	*-*awareness*, rede de conhecimento, mudança de planos de ação nos serviços.	Sensoriamento do ambiente, *fitness* aos desejos dos usuários	Manual. *Context-awareness*.	COs, RS e OBS
Autonomicidade	Amplo suporte de funcionalidades auto-*.	Autogeração de serviços (autocatálise).	Manual.	Suporte a funcionalidades auto-*.
Evolução	Autonômica. Adequação de planos de ações a mudanças de contexto e objetivos.	Autonômica. Inspirada em algoritmos genéticos. Guiada pelos usuários e recursos	Manual.	Autonômica e manual. Adequação e reusabilidade de serviços.
Interoperabilidade com Redes IP	Não IP.	IP e não IP.	IP.	IP, não IP e pós IP.

Com relação à descrição dos serviços, no GS, todas as entidades recebem um nome legível em linguagem natural, possuem um ou mais objetos descritores para descrevê-las e toda a rede é estruturada com base em ontologias publicadas.

Quanto à identificação das entidades, assim como no XIA, as entidades são identificadas de forma única e inequívoca. O identificador é obtido por meio de uma função *hash* de 512 *bits* do objeto ou através de uma impressão digital das máquinas computacionais.

No que diz respeito à descoberta, o GS utiliza uma estrutura baseada em ontologias publicadas e um Sistema de Descoberta e Busca (SDS) que permite as entidades da rede realizar a descoberta e busca semântica de serviços e conteúdos publicados. Isto elimina a sobreposição de funcionalidades presentes nas abordagens CASCADAS e XIA.

Com relação à composição dinâmica dos serviços, pode ser alcançada de forma autonômica ou de forma manual. Na forma autonômica, o OBS ou os próprios serviços possuem a capacidade de negociar com outros e se autocomporem ou comporem novos serviços. Na forma manual, o próprio usuário utiliza a invocação semântica para identificar serviços potenciais que poderão compô-lo.

Quanto à localização transparente dos serviços, no GS, cada serviço é unicamente identificado e sua localização é armazenada separadamente em um Sistema de Tabelas *Hash* Distribuídas (DHTS).

Quanto à negociação de serviços, pode ser realizada de forma autonômica por meio do OBS ou serviços capazes de representar seus interesses ou de forma manual por meio dos próprios usuários. Pode ser iniciada através da assinatura direta de descritores de serviços ou de Oportunidades de Contrato publicadas.

No que diz respeito à contratação de serviços, o contrato é formalizado por meio de um SLA e usa a forma de negociação de rodada única, sem contra ofertas. Isto é, uma parte faz a oferta de contrato, e a outra parte simplesmente aceita ou rejeita.

Com relação à confiança na prestação de serviços, além das abordagens CASCADAS e BIONETS, o GS também inclui um sistema de gestão de reputação de serviços. O Sistema de Reputação deverá refletir a trajetória das entidades, permitindo diferenciar o comportamento de cada uma conforme a QoE e QoS.

Nos quesitos adaptabilidade e autonomicidade, no GS, as entidades são proativas, cientes da situação e do contexto, capazes de aprender com suas experiências e tomar suas próprias decisões com o mínimo de interferência humana.

Quanto à evolução dos serviços, assim como as abordagens CASCADAS e BIONETS, o GS também oferece evolução autonômica. É possível orquestrar, adequar e reutilizar serviços de forma autonômica e manual.

Por fim, tem-se o requisito de interoperabilidade com as redes IP. No GS, todos os sistemas “vivem” em um ambiente virtual e pode ser utilizado junto às redes IP atuais, bem como ambientes não IP e pós IP.

Capítulo 6

Conclusões e Trabalhos Futuros

6.1 Conclusões

Neste trabalho, foram apresentados os requisitos, objetivos e ciclo de vida de entidades para uma arquitetura orientada a serviços, bem como, a forma como os serviços são negociados e contratados. Conclui-se que SOA é uma tecnologia promissora que permite criar serviços ágeis, flexíveis e capazes de se adaptar a processos de negócio dinâmicos. Esta abordagem possibilita criar uma espécie de "*self-service*" de serviços, onde todo tipo de usuário será capaz de compor serviços que resultem em aplicações mais produtivas, flexíveis, gerenciáveis e de maior rentabilidade. Algumas propostas de arquiteturas centradas em serviço e com núcleo misto, foram igualmente estudadas.

A arquitetura proposta aborda questões fundamentais para a Internet, como a identificação inequívoca de entidades e o estabelecimento do SLA entre consumidores e provedores de serviços. Além destas questões, apresenta sistemas inovadores como o OBS e RS que auxiliam as entidades no processo de orquestração de serviços. Generalizando, a arquitetura proposta permite:

- A total independência da tecnologia de interoperabilidade, podendo ser aplicada às redes IP atuais, bem como às futuras abordagens para a Internet.

- A identificação única das entidades de serviço e criação de nomes e descritores, diminuindo consideravelmente a possibilidade de existência de serviços homônimos e permitindo o aprimoramento da busca por esses serviços.

- A inserção do SLA como parte integrante da arquitetura.

- Diferentes formas de criar um serviço, tais como: manual, autonômica centralizada ou autonômica distribuída.

- Diferentes formas de iniciar processos de negociação, tais como: localização de serviços utilizando o SDS ou publicação de demanda por meio de uma CO.

- A criação de um domínio de domínios através da cooperação de diversos representantes de domínio.

- A descoberta e localização de conteúdos e serviços com base na estrutura de ontologias da rede.

- A criação de um OBS para representar os interesses dos serviços que são incapazes de negociar e contratar outros serviços, bem como realizar sua composição semântica.

- A classificação dos serviços quanto sua reputação, criando um mecanismo de evolução no ambiente digital.

- A adoção do paradigma de dois tipos de entidades habitantes na arquitetura: Informação e Computação.

- Determinar o relacionamento entre identificadores de diversas entidades, criando uma cadeia de rastreabilidade, contexto e semântica baseada em ontologias e identificadores inequívocos.

A partir dos resultados obtidos neste trabalho, pode-se afirmar que existem muitas relações entre TV digital e Internet do Futuro. Dentre as mais importantes tem-se:

- Os sistemas de TV digital podem ser expostos via representantes (*proxies*) de forma a permitir a orquestração dinâmica de seus recursos. Em outras palavras, os diversos recursos do sistema de TV digital podem ser disponibilizados na Internet do Futuro através de serviços que os representem.

- Tais serviços podem estabelecer contratos (SLAs) em nome de um dado sistema de TV digital. O próprio OBS proposto nesta dissertação poderia ser usado para este fim, representando recursos do sistema de TV digital, como por exemplo, oportunidades de anuncio, sistemas de *broadcast* de emergência, distribuição de conteúdo da Internet do Futuro na forma de *pay-per-view*, divulgação da grade de programação para serviços de autocontextualização, etc. É possível, por exemplo, orquestrar um serviço na Internet do Futuro que grave uma determinada programação da TV em função da semântica por de trás do mesmo.

- Por outro lado, os padrões de TV digital possuem recursos que podem ser usados para transportar pacotes da nova Internet.

6.2 Trabalhos Futuros

Alguns temas podem ser explorados em trabalhos futuros, como por exemplo:

- A criação e implementação de um mecanismo eficiente para gerar a identificação única das entidades de serviço.

- A criação de um *proxy* para permitir a comunicação com sistemas legados.

- A criação de mecanismos para gerenciar o SLA e mensurar a QoS.

- A criação de mecanismos ou entidades para tratar das questões de segurança e confiança entre as entidades envolvidas na negociação.

- O estudo de como realizar o roteamento de informações e serviços em ambientes autônomos e virtualizados.

- O desenvolvimento de um código fonte que implemente a arquitetura proposta, levando em consideração um ambiente altamente dinâmico, heterogêneo e escalável.

- A seleção e portabilidade para um ambiente de testes de escala global, com tráfego real e uma estrutura adequada.

Referências Bibliográficas

INTERNET WORLD STATS. **World Usage Patterns & Demographics.** Disponível em http://www.newmediatrendwatch.com/world-overview/34-world-usage-patterns-and-demographics. Último acesso em: 24 de Janeiro de 2012.

REDING, Viviane. **The Future of the Internet: A Compendium of European Projects on ICT Research Supported by the 7th Framework Programme for RTD.** European Communities. ISBN 978-92-79-08008-1. 2008.

CARDOSO Jorge, WINKLER Matthias, VOIGT Konrad, BERTHOLD, Henrike. **IoS-based services, Platform Services, SLA and Models for the Internet of Services.** University of Coimbra, Portugal. SAP Research CEC, Chemnitzer Strasse 48, 01187 Dresden, Germany. 2011.

PAPAZOGLOU Michael P., TRAVERSO Paolo, DUSTDAR Schahram, LEYMANN Frank. **Service-Oriented Computing Research Roadmap.** March 2006.

MARTINS Bruno M. **Desacoplamento de Identificadores e Localizadores: Uma Comparação de Abordagens e Proposta de Arquitetura.** INATEL – Instituto Nacional de Telecomunicações. Dissertação de Mestrado. 2011.

DUBRAY, Jean-Jacques. **Fundamentals of Service Orientation.** Attachmate White Paper. 2005.

KOTLER, Philip. **Administração de Marketing: Análise, Planejamento, Implementação e Controle.** Livro. 5ª ed. São Paulo, Brasil. Editora Atlas. 1998.

CARTER, Sandy. **The New Language of Business: SOA & Web2.0.** Book. The Rational Edge, Editorial Staff, IBM. ISBN 013195654X. February 2007.

ENDREI, Mark, ANG Jenny, ARSANJANI Ali, CHUA Sook, COMTE Philippe, KROGDAHL Pal, LUO Min, NEWLING Tony. **Patterns: Service-Oriented Architecture and Web Services.** IBM Redbooks. First Edition. April 2004.

AIELLO Marco, DUSTDAR Schahram. **Service-Oriented Computing: Service Foundations.** TUWien. Dagstuhl Seminar Proceedings 05462 – Service Oriented Computing (SOC). 2006.

ROSEN Michael, LUBLINSKY Boris, SMITH T. Kevin, BALCER J. Marc. **Applied SOA: Service-Oriented Architecture and Design Strategies.** Wiley Publishing, Inc. 2008.

SOA PRINCIPLES. **An Introduction to the Service-Orientation Paradigm.** Disponível em http://www.soaprinciples.com/. Último acesso em: 16 de Janeiro de 2012.

CHANNABASAVAIAH Kishore, HOLLEY Kerrie, TUGGLE Edward M. **Migrating to a Service-Oriented Architecture.** IBM Corporation. Software Group. United States of America. April 2004.

GÜNER Shelda. **Architectural Approaches, Concepts and Methodologies of Service Oriented Architecture.** Master Thesis. TUHH – Technische Universitãt Hamburg-Hamburg. Hamburg, Germany. August 2005.

LALIWALA, Zakir. **Event-driven Service-oriented Architecture for Dynamic Composition of Web Services.** Thesis Doctor of Philosophy in Information and Communication Technology. DA-IICT - Dhirubhai Ambani Institute of Information and Communication Technology, Gandhinagar. 2007.

PAPAZOGLOU Mike P., HEUVEL Willen-Jan van den. **Service Oriented Architecture: approaches, technologies and research issues.** The VLDB Journal. ISBN: 106688880949877X. 2007.

MODI Gurpreet S. **Service Oriented Architecture & Web 2.0.** Seminar. Department of Computer Science and Engineering Guru Tegh Bahadur Institute of Technology, New Delhi. 048/GTBIT/CSE/2004. 2004.

ELFATATRY Ahmed, LAYZELL Paul. **Negotiating in Service-Oriented Environments.** Communications of the ACM, vol. 47, nº. 8. August 2004.

MANZALINI Antonio, MANNELLA Antonietta, MOISO Corrado, HASELOFF Sandra, KUSBER Rico, BRGULJA Nermin, ZAMBONELLI Franco, DEUSSEN Peter H., SAFFRE Fabrice, BARESI Luciano, LENT Ricardo, GELENBE Erol, FERDINANDO Antonio D., CASCELLA Roberto. **CASCADAS - Bringing Autonomic Services to Life**. Deliverable 8.4. January 2009.

HASELOFF Sandra, KUSBER Rico, BRGULJA Nermin, BENKŐ Borbála K., DEUSSEN Peter, HÖFIG Edzard, MANZALINI Antonio, ALFANO Rosario, MANNELLA Antonietta, GIACOMETTO Mario, MAMEI Marco. **CASCADAS - First Prototype Integration (release 1).** Deliverable 1.3. January 2008.

HOFIG H., BENKŐ B. K., DI Nitto E., MAMEI M., MANNELLA A. and Wuest B. **On Concepts for Autonomics Communication elements.** In Proc. Of the 1st IEEE International Workshop on Modeling Autonomic Communication Environments (MACE 2006), Dublin, October 2006.

MARROW P., KOUBARAKIS M., VAN LENGEN R.H., VALVERDE-ALBACETE F., BONSMA E., CIDSUERIO J., FIGUEIRAS-VIDAL A. R., GALLARDO-ANTOLIN A., HOILE C., KOUTRIS T., MOLINA-BULLA H., NAVIA-VASQUEZ A., RAFTOPOULOU P., SKARMEAS N., TRYFONOPOLOULOS C., WANG F. and XIRUHAKI C. **Agents in Decentralised Information Ecosystems: the DIET Approach.** In Proc. of

the Artificial Intelligence and Simulation Behaviour Convention (AISB 2001). York, March 2001.

BENKŐ, Borbála K., HÖFIG Edzard, BRGULJA Nermin, KUSBER Rico. **Adaptive Services in a Distributed Environment.** IEEE. Conference Location: Kassel. ISBN: 978-0-7695-3389-6. December 2008.

BAUMGARTEN, Matthias, ZAMBONELLI Franco, CASTELLI Gabriella, BICOCCHI Nicola, KUSBER Rico, BRGULJA Nermin. **Final Report on Open Toolkit for Knowledge Networks and on Advanced Knowledge Networks Concepts and Models.** Deliverable 5.4. December 2008.

MANZALINI Antonio, MANNELLA Antonietta, ALFANO Rosario, HÖFIG Edzard, MAMEI Marco, BENKŐ Borbála K., KATONA Tamas, KUSBER Rico, BRGULJA Nemin. **Bringing Autonomic Services to Life: Report on state-of-art, requirements and ACE model.** Deliverable 1.1. January 2007.

BIONETS. **Overview: BIOlogically-Inspired Networks and Service**. Disponível em http://www.bionets.eu/index.php?area=11. Último acesso em: 21 de Julho de 2011.

HUEBSCHER C. Markus and McCANN A. Julie. **A Survey of Autonomic Computing - Degrees, Models, and Applications.** Imperial College London. ACM Journal Name, Vol. V. August 2008.

LINNER David, PELLEGRINI Francesco D., MIORANDI, Daniele, MOISO Corrado, BACSARDI Laszlo. **BIONETS Architecture: from Networks to SerWorks.** IEEE. Conference Location: Budapest. E-ISBN: 978-963-9799-05-9. December 2007.

TAHKOKORPI Markku, LATVAKOSKI Juhani, BORGIA Eleonora, PANAGAKIS Antonis, SIMON Vilmos, PELLEGRINI Francesco D., HAUTAKOSKI Tomi, SCHRECKLING Daniel, MIORANDI Daniele. **BIONETS Requirements and Architectural Principles: Architecture, Scenarios and Requirements Refinements.** Deliverable 1.1.1. August 2006.

TSCHUDIN C. and Yamamoto L. **Self-evolving network software.** Praxis der Informationsverarbeitung und Kommunikation (PIK Magazine), pp. 206-210. K.G. Saur Verlag, Munich, Germany. December 2005.

MIORANDI Daniele, Yamamoto Lidia, Dini Paolo. **Service evolution in bio-inspired communication systems**. European Commision. Project: BIONETS. Proceedings of the International Conference on Self-Organization and Autonomous Systems in Computing and Communications (SOAS 2006). ITSSA – International Transactions on Systems Science and Applications Journal. Vol. 2, Nº. 1, pp. 51-60. September 2006.

PELLEGRINI Francesco D., MIORANDI Daniele, CARRERAS Iacopo, COHEN Dany R. R., LATVAKOSKI Juhani, HAUTAKOSKI Tomi, YAMAMOTO Lidia, NEGLIA Giovanni, ALOUF Sara, SCHRECKLING Daniel, PELUSI Luciana, BORGIA Eleonora, PETROCCHI Marinella, MARTINELLI Fabio, MOISO Corrado, MANZALINI Antonio. **BIONETS Infrastructure and Design: Disappearing Network Autonomic Operation and Evolution.** Deliverable 1.2.2. August 2007.

ANAND Ashok, DOGAR Fahad, HAN Dongsu, LI Boyan, LIM Hyeontaek, MACHADO Michel, WU Wenfei, AKELLA Aditya, ANDERSEN Michel, BYERS John, SESHAN Srinivasan, STEENKISTE Peter. **XIA: An Architecture for an Evolvable and Trustworthy Internet.** January 2011.

eXpressive Internet Architecture. Disponível em http://www.cs.cmu.edu/~xia/technical/technical-approach.html. Último acesso em: 02 de Dezembro de 2011.

VAZ Agostinho M. **Internet de Serviços: Estudo, Análise e Proposta de Arquitetura Conceitual.** Dissertação de Mestrado. INATEL – Instituto Nacional de Telecomunicações. 2011.

MICROSOFT. **Grupos de trabalho em comparação com domínios.** Disponível em http://technet.microsoft.com/pt-br/library/cc739052(WS.10).aspx. Último acesso em: 25 de Agosto de 2011.

WIKIPEDIA. **Distributed Hash Table**. Disponível em http://pt.wikipedia.org/wiki/ Distributed_hash_table. Último acesso em: 09 de Agosto de 2011.

LIU Shi, BI Jun, WANG Yangyang. **A DHTs-Based Mapping System for Identifier and Locator Separation Network.** IEEE. Conference Location: Athens. ISBN: 978-0-7695-3664-4. June 2009.

CIRANI Simone & VELTRI Luca. **Implementation of a framework for a DHT-based Distributed Location Service.** Dpt. Information Engineering, University of Parma, Parma, Italy. 2007.

STOICA Ion, MORRIS Robert, LIBEN-NOWELL David, KARGER David, KAASHOEK M. Frans, DABEK Frank, BALAKRISHNAN Hari. **Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications.** IEEE/ACM Transactions on Networking. ISSN: 1063-6692. February 2003.

MAYMOUNKOV Petar and MAZIÈRES David **Kademlia: A Peer-to-Peer Information System Based on the XOR Metric.** In Processings of the IPTPS, Cambridge, MA, USA. February 2002.

ZHAO B. Y., Huang L., STRIBLING J., RHEA S. C., JOSEPH A. D., and KUBIATOWICZ J. D. **Tapestry: A resilient global-scale overlay for service deployment**. IEEE. Journal on Selected Areas in Communications, vol. 22, nº 1. January 2004.

ROWSTRON A. and DRUSCHEL P. **Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems.** In Proceedings of the Middleware. 2001.

GHODSI Ali. **Distributed k-ary System: Algorithms for Distributed Hash Tables.** School of Information and Communication Technology. Department of Electronic, Computer, and Software Systems. Stockholm, Sweden. December 2006.

MILOJICIC S. Dejan, KALOGERAKI Vana, LUKOSE Rajan, NAGARAJA Kiran, PRUYNE Jim, RICHARD Bruno, ROLLINS Sami, XU Zhichen. **Peer-to-Peer Computing.** HP Laboratories Palo Alto. HPL-2002-57. July 2003.

JACOBSON Van, SMETTERS Diana K., THORNTON James D., PLASS Michael F. BRIGGS Nicholas H., BRAYNARD Rebecca L. **Networking Named Content.** Palo Alto Research Center (PARC). ACM CoNEXT. Rome, Italy. December 2009.

RAHRER Tim., FIANDA Riccardo., WRIGHT Steven. **Triple-Play Services Quality of Experience (QoE) Requirements.** DSL Forum – Technical Report (TR-126). December 2006.

PELLISSARI Felipe R., RIGHI Rafael, WESTPHALL Carla M. **RBRP: Protocolo de Reputação Baseado em Papéis para Redes Peer-to-Peer.** V Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais. Florianópolis, Santa Catarina, Brasil. 2005.

AHO Alfred V., LAM Monica S.; SETHI Ravi. **Compiladores: Princípios, técnicas e ferramentas.** Tradução de Daniel Vieira; Revisão de Mariza Andrade da Silva Bigonha. 2. ed. São Paulo: Pearson Addison-Wesley. 2008.

CIFUENTES Cristina. **Reverse Compilation Techniques.** Thesis. Doctor of Philosophy. Queensland University of Technology. July 1994.

WISEGEEK**. What is Modular Programming?** Disponível em http://www.wisegeek.com/what-is-modular-programming.htm. Último acesso em: 16 de Novembro de 2011.

SOMMERVILLE, Ian. **Engenharia de software**. Tradução: Maurício de Andrade. 6 ed. São Paulo: Person Education. 2004.

JABUR, Wilson P. **Engenharia Reversa de Software.** Congresso sobre Direito de Autor e Interesse Público. Maio de 2007.

ANDRIEUX, Alain, CZAJKOWSKI Karl, DAN Asit, KEAHEY Kate, LUDWIG Heiko, NAKATA Toshiyuki, PRUYNE Jim, ROFRANO John, TUECKE Steve, XU Ming. **Web Services Agreement Specification (WS-Agreement).** Grid Resource Allocation Agreement Protocol (GRAAP). March 2007.

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

MIORANDI Daniele, DINI Paolo, ALTMAN Eitan, KAMEDA Hisao. **Paradigm Applications and Mapping: Framework for Distributed On-line Evolution of Protocols and Services, 2nd Edition**. Deliverable 2.2.2. June 2007.