GENESIS SYSTEM: PROPOSTA DE UMA ARQUITETURA PARA ORQUESTRAÇÃO DE SERVIÇOS EM AMBIENTES AUTONÔMICOS ORIENTADOS A SERVIÇO

Rodrigo Carneiro Brandão, Instituto Nacional de Telecomunicações – INATEL, E-mail: rodrigocarneirobrandao@hotmail.com
Antônio Marcos Alberti, Instituto Nacional de Telecomunicações – INATEL, E-mail: alberti@inatel.br

Resumo: Neste artigo, abordamos uma Internet orientada a serviços, onde "tudo é visto como serviço". São apresentadas três propostas de redes futuras centradas em serviços e de núcleo misto, centradas em *hosts*, serviços e conteúdos. Por fim, é proposto o *Genesis System*, uma arquitetura conceitual caracterizada pelo fraco acoplamento, reusabilidade e a possibilidade de orquestrar serviços de forma manual e autonômica. Entre as inovações desta proposta, destacam-se o *Reputation System*, que cria um mecanismo de evolução no ambiente digital e o *Orchestration Broker System*, responsável por orquestrar serviços que atendam demandas específicas e representar entidades de serviços incapazes de se auto-representarem. As características fundamentais da proposta são a total independência da tecnologia de interoperabilidade, o alto grau semântico e a identificação inequívoca de entidades de serviço que podem ser orquestradas em tempo de execução para atender de forma ágil e adequada as necessidades de processos de negócios específicos, complexos e dinâmicos.

Palavras Chave: Serviços, orquestração, ambientes orientados a serviços, oportunidades de contrato e SLA.

GENESIS SYSTEM: AN ARCHITECTURE PROPOSAL TO ORCHESTRATE SERVICES ON AUTONOMIC SERVICE-ORIENTED ENVIRONMENTS

Abstract: In this paper, we address a service-oriented Internet architecture, where "everything is seen as a service". We present three proposals for service-, content-, and host-centric future networks. We also propose a conceptual architecture based on a Genesis System where components are loosely coupled, reusable, and manually or autonomously orchestrated. Among the inovations behind this proposal, we highlight a reputation management system, which envision the digital evolution of entities inside the architecture, as well as an Orchestration Broker System, which aims to orchestrate services targeted to specific demands and to advocate for small services unable to self-represent. The proposed architecture is agnostic regarding the underlying communication technologies. Also, service orchestration is based on rich semantics information, aligned with business requirements. In addition, services have self-certifying names that can be used to unequivocally identify them in execution time.

Keywords: Services, orchestration, service-oriented environments, contract opportunities, service-level agreements.

1. INTRODUÇÃO

Desde a origem da Internet, esta tem crescido e se popularizado rapidamente, com uso cada vez mais diversificado. O modelo da Internet atual foi projetado com base no paradigma *host-centric*, que colocou os terminais da rede no núcleo do projeto original, deixando-o simples e concentrando toda a inteligência nas funcionalidades dos sistemas finais. Este modelo facilitou e acelerou o crescimento da Internet, uma vez que não é necessário alterar o núcleo da rede para a criação de novas aplicações e sistemas. Em contrapartida, este paradigma praticamente "engessa" a arquitetura, tornando difícil resolver problemas como escalabilidade, flexibilidade e confiabilidade. Infelizmente, uma alternativa comum a estes problemas tem sido desenvolver soluções paralelas que são "remendadas" à arquitetura atual, dificultando ou até mesmo impossibilitando soluções mais expressivas.

A transição de uma economia global baseada em produtos para uma economia baseada em serviços é uma tendência no mercado de negócios e tem sido refletida na Internet. O mundo físico está convergindo com o mundo digital. Entretanto, os "remendos" e os diversos problemas ainda sem solução, impedem que a Internet atual atenda a estas novas demandas de forma adequada e satisfatória.

Neste contexto, propomos uma Internet de Serviços (IoS – *Internet of Service*), que possibilitará a todo tipo de usuário modelar, prover, comercializar e consumir serviços disponíveis na Internet de forma simples, ágil e flexível [CARDOSO, et. al., 2011]. Na IoS tudo é visto como serviço: *software*, plataformas computacionais e infraestrutura.

O conceito de "tudo como serviço" nos remete ao paradigma de *Service Oriented Computing* (SOC). Na SOC o serviço é o elemento básico para o desenvolvimento e composição de aplicações distribuídas, mesmo em ambientes heterogêneos [PAPAZOGLOU, et. al., 2006].

A SOC é a base de uma *Service-Oriented Architecture* (SOA). Um estilo de arquitetura de *software* cujo princípio fundamental é que as funcionalidades implementadas pelas aplicações sejam disponibilizadas na forma de serviços independentes com interfaces bem definidas, que podem ser invocadas em sequências definidas para formar processos de negócio [CHANNABASAVAIAH, et. Al., 2004]. Estes serviços são autônomos, fracamente acoplados, independentes de plataforma computacional e de linguagem de programação, podendo ser reutilizados e combinados para permitir um melhor alinhamento entre a Tecnologia de Informação (TI) e os negócios.

Neste sentido, propomos uma arquitetura que visa não apenas compor serviços, mas sim, estudar formas de orsquestrá-los autonomicamente. Para que a orquestração de serviços autônomos seja possível, a arquitetura conta com uma estrutura básica de sistemas: *Domain System* (DS), *Public Subscribe System* (PSS), *Distributed Hash Table System* (DHTS), *Generic Indirection Resolution System* (GIRS) e *Search and Discovery System* (SDS).

DS: É a entidade responsável por representar os interesses do domínio perante outros domínios, conduzindo processos de busca e seleção de domínios parceiros e estabelecendo contratos de cooperação. A parceria estabelecida entre vários representantes de domínio resulta na criação de um domínio de domínios e assim por diante.

PSS: 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. As publicações e assinaturas podem ser realizadas de duas formas: com notificação ou sem notificação. A primeira ocorre quando faz-se 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)*. A segunda é quando não há a necessidade de notificar a outra entidade, e possui a seguinte estrutura: *Pub/Sub (Chave; <Mapeamento>, Tipo***)**. Quanto as estruturas, a Chave refere-se a identificação da entidade publicadora ou assinante, o ID-Notificação refere-se a identificação da entidade que será notificada pelo PSS; e por fim, o Tipo é o número do mapeamento que possibilitará diferenciar publicações e assinaturas de conteúdos e serviços.

Distributed Hash Tables System: As DHTs (*Distributed Hash Tables*) são uma classe de sistemas distribuídos descentralizados que oferecem 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 [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].

GIRS: É um sistema proposto por [MARTINS, 2011], para tratar indireções utilizadas em propostas de Internet do futuro, em especial nas redes centradas em informação [JACOBSON, et. al., 2009]. Atua como um catálogo de endereços, mapeando de forma inequívoca identificadores distribuídos à suas respectivas entidades, criando assim uma cadeia de rastreabilidade, contexto e

semântica baseada em uma estrutura de ontologias da rede. Para que o GIRS funcione corretamente, todas as entidades (reais e virtuais) devem ser identificadas de forma única; receber um nome legível em linguagem natural; e possuir um ou mais objetos descritores para descrevê-las.

SDS: Possibilita as entidades realizar consultas e recuperar conteúdos e serviços, considerando não apenas padrões estruturais e sintáticos, mas também padrões semânticos. 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.

O restante deste artigo é dividido da seguinte forma: Na seção II serão apresentadas três importantes abordagens de redes centradas em serviço e de núcleo misto: CASCADAS (*Component-ware for Autonomic Situation-aware Communications, and Dynamically Adaptable Services*), BIONETS (*BIOlogically-inspired NETworks and Services*) e XIA (*eXpressive Internet Architecture*). Na seção III contém a principal contribuição deste artigo, o *Genesis System*, responsável pela instanciação e criação dos demais sistemas e entidades da arquitetura, bem como uma comparação com os demais trabalhos relacionados. Finalmente, na seção IV serão apresentadas as conclusões do artigo.

2. ABORDAGENS RELACIONADAS

Nas pesquisas realizadas para o estudo e proposta da arquitetura foram identificados alguns trabalhos relacionados. Esta seção apresenta uma visão geral dos projetos: CASCADAS, BIONETS e XIA.

A. CASCADAS

O *Component-ware for Autonomic Situation-aware Communications, and Dynamically Adaptable Services* (CASCADAS) é um projeto do *Sixth Framework Programme* (FP6). 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.

O projeto CASCADAS tem como objetivo 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. 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 para proporcionar a funcionalidade desejada.

a) ACE – Autonomic Communication Elements

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 [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". A seguir é descrito 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 não orientado à conexão (*GN - Goal Needed*/*GA – Goal Achievable*) [HOFIG, et. al., 2006] é usado para descobrir os serviços, através do paradigma publica/assina 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. Já 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.
- **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.*

b)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].

O nível baixo da KN 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

tornar-se 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.

B. BIONETS

O *BIOlogically-inspired NETworks and Services* (BIONETS) é um projeto do *Sixth Framework Programme* (FP6).

O projeto BIONETS tem como objetivo 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.

a) Arquitetura

BIONETS descreve uma arquitetura para sistemas de computação autonômica que pode ser dividida em três grandes partes apresentados a seguir.

- **Framework de Serviços:** Possibilita a execução dos serviços nos nós, e, por meio de estratégias de adaptação e evolução, permite que os serviços reajam de forma autonômica às mudanças no ambiente. Este tipo de autonomicidade pode ser suportada a nível de nó e de serviço. A nível de nó tem melhor desempenho em relação à complexidade e escalabilidade, embora a autonomicidade a nível de serviço tenha uma maior flexibilidade. Para possibilitar uma maior flexibiliade a nível de nó, foram criados os mediadores de serviços. Entidades capazes de interagir com os serviços por meio dos modelos de interação disponibilizados pelo Framework de Interação, mesmo que eles estejam hospedados em nós diferentes [LINNER, et. al., 2007].
- **Framework de Interação:** Disponibiliza modelos de interação que possibilitam desacoplar o Framework de Serviços dos protocolos de comunicação subjacentes, nomeando e endereçando esquemas e características da rede. Os principais modelos relacionados por [LINNER, et. al., 2007] são: *Semantic Data Space* (SDS), *Distributed Hash Tables* (DHT) e o paradigma *Publish/Subscribe*.
- **Framework 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.

BIONETS é construída em cima de uma arquitetura de rede de duas camadas. 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. Além dos elementos citados, esta arquitetura pode conter um terceiro elemento, o AP (*Acess Point*), que possibilita a interoperabilidade entre ambientes IP e BIONETS.

b) 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.

A evolução do serviço implica em uma adaptação de longo prazo para as mudanças no ambiente, podendo os serviços adquirem novas funcionalidades perante o sistema [TSCHUDIN, et. al., 2005] e [MIORANDI, et. al., 2006]. Foram investigadas as seguintes ordens de evolução de serviços:

- **Evolução de Ordem "0"**: É 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. Em cada nó, um genótipo[1] descreve o esquema de encaminhamento usado, um processo de seleção favorece a difusão dos genótipos 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].
- **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 [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" [2], que poderão ser injetados em um ambiente adequado para se tornarem programas a serem executados [LINNER, et. al., 2007].

C. 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).

XIA tem como objetivo 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.

[1] **Genótipo** na área de Algoritmos é a estrutura ou representação interna usada para armazenar os parâmetros a serem otimizados.
[2] **Gene** é a unidade fundamental da hereditariedade.

Se comparado com as demais abordagens relacionadas, a distinção mais visivel, é que o núcleo do XIA aborda diferentes modelos e é totalmente flexível, não sendo fixa a utilização de uma base específica (hosts, por exemplo). XIA permite que soluções da arquitetura atual (centrada em *hosts*) sejam aliadas a novas proposta de arquitetura (centradas em serviço e conteúdo, por exemplo) permitindo uma maior flexibilidade de suas entidades.

a) Arquitetura

O projeto é estruturado com base em três pilares [ANAND, et. al., 2011]: *principal-centric*, *fallbacks* e identificadores intrinsecamente seguros.

- ***Principal-centric*** (*hosts*, conteúdos e serviços): Deve possibilitar os usuários e aplicações expressarem sua intenção, dando à rede uma flexibilidade significativa, acrescentando e/ou modificando modelos que compõem o núcleo da arquitetura com uma complexidade razoavelmente baixa.
- ***Fallbacks*:** Permite as entidades especificarem ações alternativas caso os roteadores não operem sobre a intenção primária. 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 alternativa (identificador de *hosts*) para recuperar o conteúdo desejado a partir de um servidor conhecido [ANAND, et. al., 2011].
- **Identificadores Intrinsecamente Seguros**: Possibilita as entidades validarem que estão se comunicando com o *principal-centric* correto, bem como, diferenciar suas informações semânticas.

Além dos três pilares já destacados, o núcleo da arquitetura contém 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 identificador único e (iii) um método para mapeá-lo. Os identificadores e nomes de objetos utilizados em rede 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 [ANAND, et. al., 2011].

3. SISTEMA GENESIS

Nesta seção, apresenta-se a principal contribuição deste artigo, o *Genesis System* (GS), responsável pela instanciação de todos os demais sistemas (DHTS, PSS, GIRS, SDS e DS). Além da 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 uma proposta de arquitetura para a Internet do Futuro em concepção no INATEL. A arquitetura na qual o GS oferece o serviço de instanciação e criação 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 1** ilustra tais habitantes no contexto do Sistema Genesis.

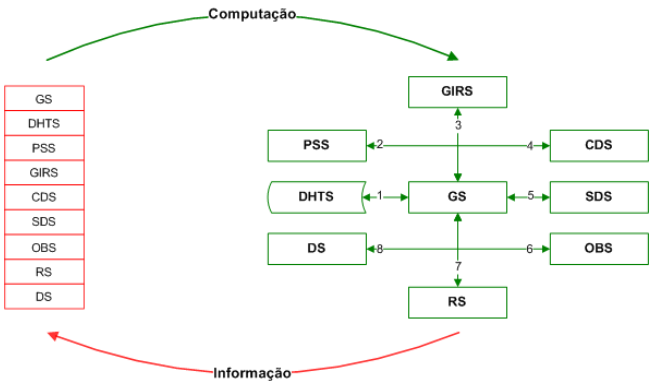

Figura 1: **GS – Informação e Computação**

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.

Observe ainda na **Figura 1**, que foram instanciados ordenadamente os sistemas básicos da arquitetura que compõe o domínio e em seguida o DS que irá representar os interesses de todos os habitantes 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 compor-lo. Na segunda, o OBS através da auto-organização semântica compõe novos serviços para atender alguma demanda específica do serviço do usuário. 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.

Generalizando a composição de serviços, esta pode ser realizada de duas formas: através da busca semântica direta de descritores de serviços ou por meio da publicação de *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 realiza uma busca semântica direta por descritores de serviços de seu interesse. Se houver resultados condizentes, estes são retornados para a entidade na forma de descritores e/ou nomes legíveis. Então, 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, 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. Então, a entidade elege as COs condizentes com sua área de especialização, e em seguida avança para 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, 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 aos habitantes do domínio tomar conhecimento de sua existência.

A.Sistemas Criados pelo Genesis System

Para melhor entendimento da proposta, são apresentados a seguir os sistemas criados pelo GS e que estão diretamente relacionados com o processo de composição de serviços.

a)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 OBS é constituído por funcionalidades comuns e específicas. A funcionalidade comum diz respeito à sua atividade primária, que é auxiliar os serviços de usuários incapazes de negociar e contratar serviços, representando seus interesses e realizando sua composição semântica. A funcionalidade específica refere-se à área de negociação ou especialização associada a cada entidade OBS. A **Figura 2** ilustra o processo de seleção do OBS conforme funcionalidade específica.

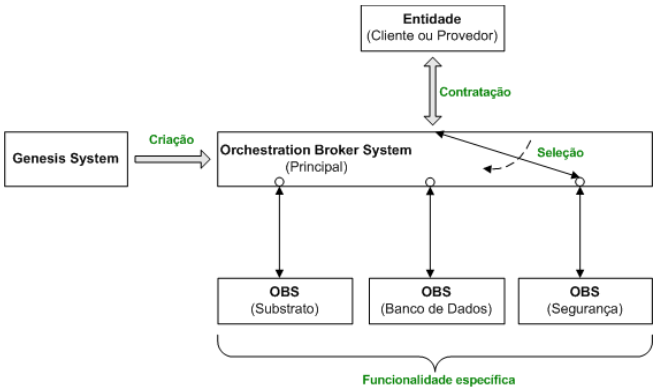

Figura 2: *OBS associado a algumas áreas de especialização possíveis.*

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.

Dado que o OBS foi criado e está operando, seus serviços podem ser contratados de duas formas: manual e autonômica. Manual, quando o usuário junto ao OBS define como deverá ser realizada a representação do seu serviço. Autonômica, quando 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. O contrato deverá ser formalizado por meio de um *Service Level Agreement* (SLA).

Considerando que o SLA foi estabelecido, o OBS deverá se relacionar de maneira cooperativa com todas as entidades da rede e vice-versa. Por meio da cooperação, o OBS pode representar os interesses de seus contratantes, negociando seu fornecimento ou composição de novos serviços junto a outras entidades da rede. Além da cooperação, o OBS compartilha questões de competição, que podem existir com outras entidades que sejam capazes de negociar seus próprios interesses.

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.

b) 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.

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 - *Quality of Experience*), devendo a avaliação objetiva (QoS - *Quality of Service*) ser mensurada de alguma forma e 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].

c) CDS – Compilation and Decompilation System

O CDS realiza a compilação e descompilação de entidades de serviço. O módulo de compilação é responsável por compilar códigos fonte (entidades no estado de informação) criando entidades de 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). Já 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, sendo vedado às demais entidades fazer engenharia reversa ou descompilação de entidades de serviço.

B. Mapeamentos para a Arquitetura

Nesta seção, utiliza-se a flexibilidade de mapeamentos GIRS para apresentar as relações ilustradas nas Figuras 2 e 3. Os mapeamentos são implementados através de registros em *hash* distribuídos. A Tabela 1 contém alguns mapeamentos para a arquitetura proposta com as suas chaves de entrada e valores de saída.

Mapeamento	**Chave**	**Valor (XML)**	
		Tipo	**Descrição**
<ID-Nome; ID-Servico>	ID-Nome	1	Lista de serviços que possuem exatamente o mesmo nome.
<ID-Nome; ID-Descritor>	ID-Nome	2	Lista dos descritores de objetos de serviços associados ao ID-Nome.
<ID-Servico; ID-Substrato>	ID-Serviço	3	Lista de identificadores de substrato onde se encontra executando ou armazenado o serviço requisitado.
<ID-SLA; ID-Serviço>	ID-SLA	4	Lista dos identificadores de serviços que fazem parte do SLA com este ID.
<ID-Entidade; ID-Reputação>	ID-Entidade	5	Lista dos identificadores de reputação associados a uma determinada entidade.
<ID-Nome; ID-CO>	ID-Nome	6	Lista de COs que possuem exatamente o mesmo nome.
<ID-CO; ID-Substrato>	ID-CO	7	Lista de Substratos que possuem armazenado o conteúdo de uma determinada CO.

Tabela 1: Exemplos de mapeamentos acomodados no GIRS.

A Tabela 1 relaciona 7 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 2 e 3.

C. Localização e Contratação de Serviços

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 e vários domínios. Neste contexto, todo o relacionamento entre as entidades é baseado no 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.

a) Localização e Contratação de Serviços em um Domínio

Considere o estudo de caso ilustrado pela **Figura 3**, 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.

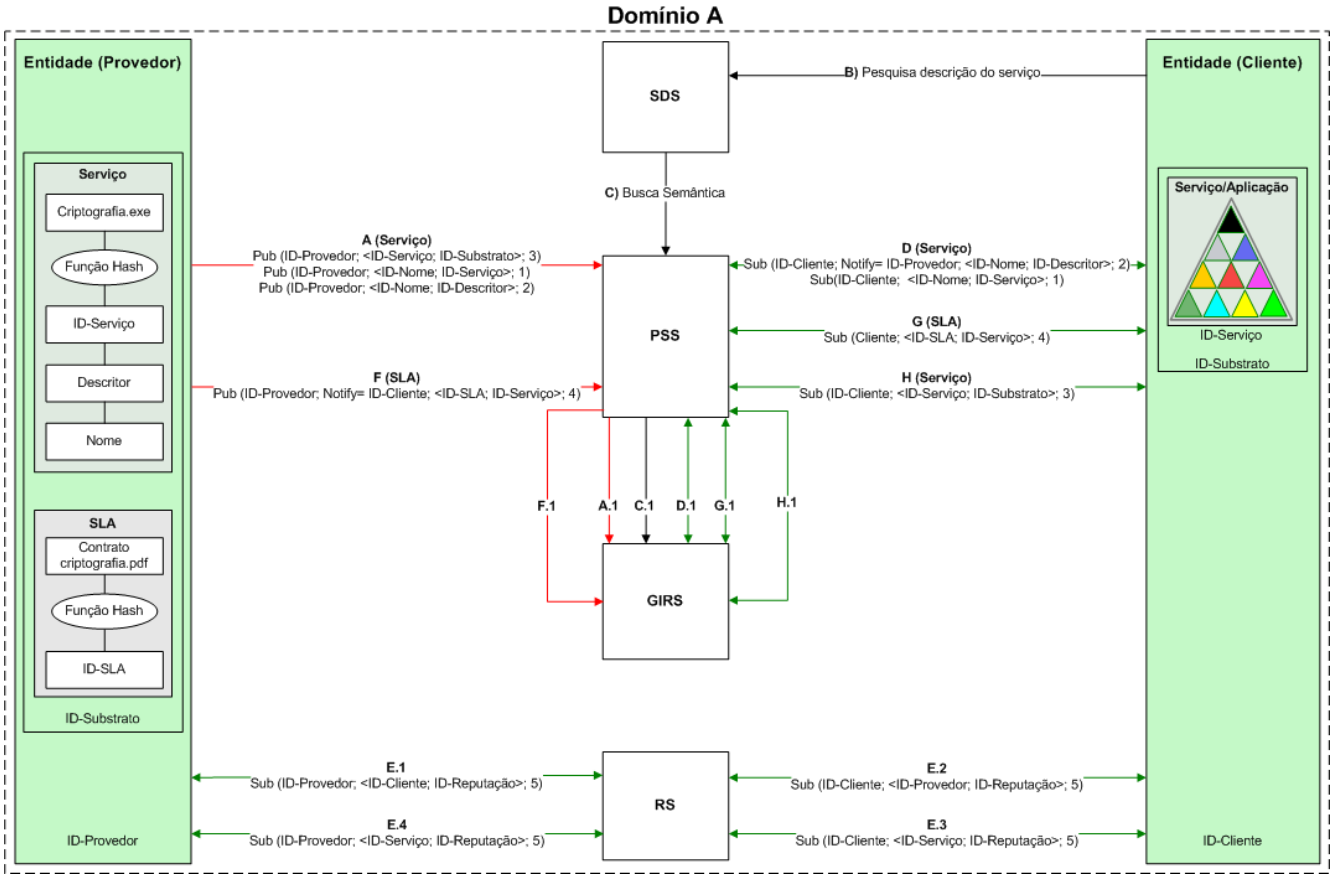

Figura 3: *Localização e contratação de serviço em um domínio.*

Após estas interações, a entidade responsável pelo serviço (Provedor) deve publicar no PSS o serviço (*Criptografia.exe*) que oferece (mapeamento 3 da **Tabela 1**), o nome do serviço (mapeamento 1 da **Tabela 1**) e seu descritor (mapeamento 2 da **Tabela 1**), 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 2 e 1 da **Tabela 1**, representados pelo passo D).

Após ser notificado, o Provedor por meio do RS verifica a reputação do Cliente e vice-versa (mapeamento 5 da **Tabela 1**, representado pelos passos E.1 e E.2), bem como o cliente verifica a reputação do serviço que deseja contratar (mapeamento 5 da **Tabela 1**, 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 5 da **Tabela 1**, representado pelo passo E.4). São realizadas inúmeras interações até que haja um consenso entre as entidades envolvidas (etapa não ilustrada na **Figura 3**).

A contratação do serviço é formalizada por meio de um SLA (*Contrato Criptografia.pdf*). 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 4 da **Tabela 1**, representado pelo passo F), possibilitando a este assinar o identificador do SLA (mapeamento 4 da **Tabela 1**, 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 3 da **Tabela 1**, representado pelo passo H).

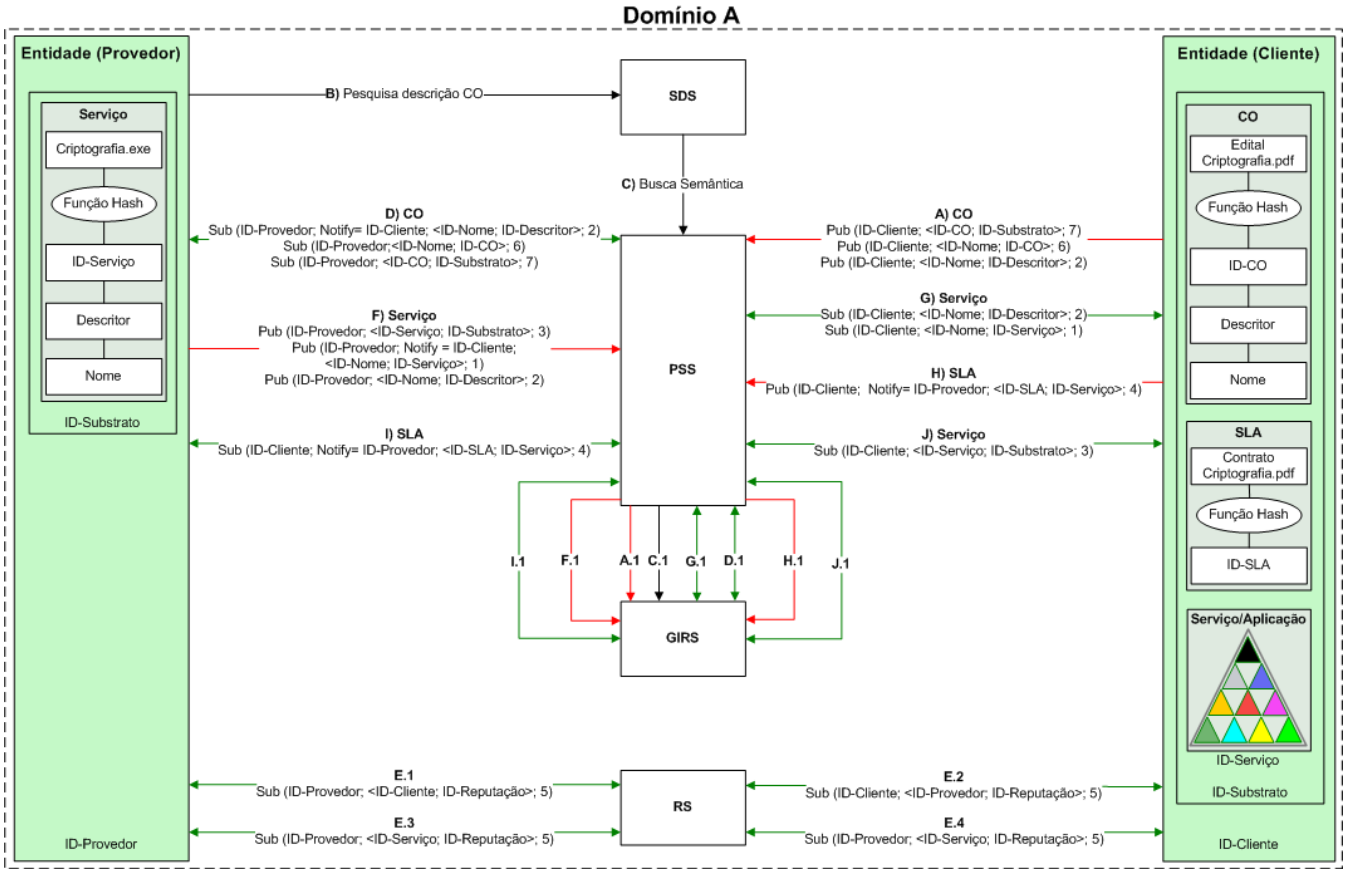

Figura 4: *Publicação de CO e contratação de serviço em um domínio.*

Na **Figura 3**, a entidade Cliente realizou uma busca por um serviço específico que já existia no domínio, sendo este localizado, negociado e contratado. Já na **Figura 4**, parte-se do pressuposto que o Cliente não encontrou um serviço condizente com sua pesquisa, sendo necessário publicar uma Oportunidade de Contrato (*Edital Criptografia.pdf*) com o intuito de despertar o interesse de alguma entidade em criar o serviço solicitado.

O Cliente responsável pela Oportunidade de Contrato (*Edital Criptografia.pdf*) deve publicar no PSS o conteúdo da CO (mapeamento 3 da **Tabela 1**), o nome da CO (mapeamento 6 da **Tabela 1**) e seu descritor (mapeamento 2 da **Tabela 1**), (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 2, 6 e 7 da **Tabela 1**, representados pelo passo D).

Após ser notificado, o Cliente por meio do RS verifica a reputação do Provedor e vice-versa (mapeamento 5 da **Tabela 1**, 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 5 da **Tabela 1**, 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 4**).

Caso o serviço solicitado ainda não exista, o Provedor deverá desenvolver, compilar, identificar e publicar o novo serviço (*Criptografia.exe*) no PSS (mapeamento 3 da **Tabela 1**), o nome do serviço, bem como notifica o cliente da publicação (mapeamento 1 da **Tabela 1**), e por fim, publicar o descritor do serviço (mapeamento 2 da **Tabela 1**), (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 2 e 1 da **Tabela 1**, representado pelo passo G) e formaliza a contratação do serviço por meio de um SLA (*Contrato Criptografia.pdf*). Isto é, o Cliente por meio do PSS publica o identificador do SLA no GIRS e notifica o Provedor sobre sua publicação (mapeamento 4 da **Tabela 1**, representado pelo passo H). Então, o Provedor assina o identificador do SLA e notifica o cliente sobre sua assinatura (mapeamento 4 da **Tabela 1**, 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 3 da **Tabela 1**, representado pelo passo J).

Observe que o processo de publicação e assinatura de conteúdos e serviços é similar, podendo os sistemas ilustrados nas **Figuras 3** e **4** 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.

b) Localização e Contratação de Serviços em vários Domínios

Diferentemente dos cenários apresentados até o momento, a **Figura 5** ilustra a contratação de serviços por meio do SDS e publicação de COs em diversos domínios.

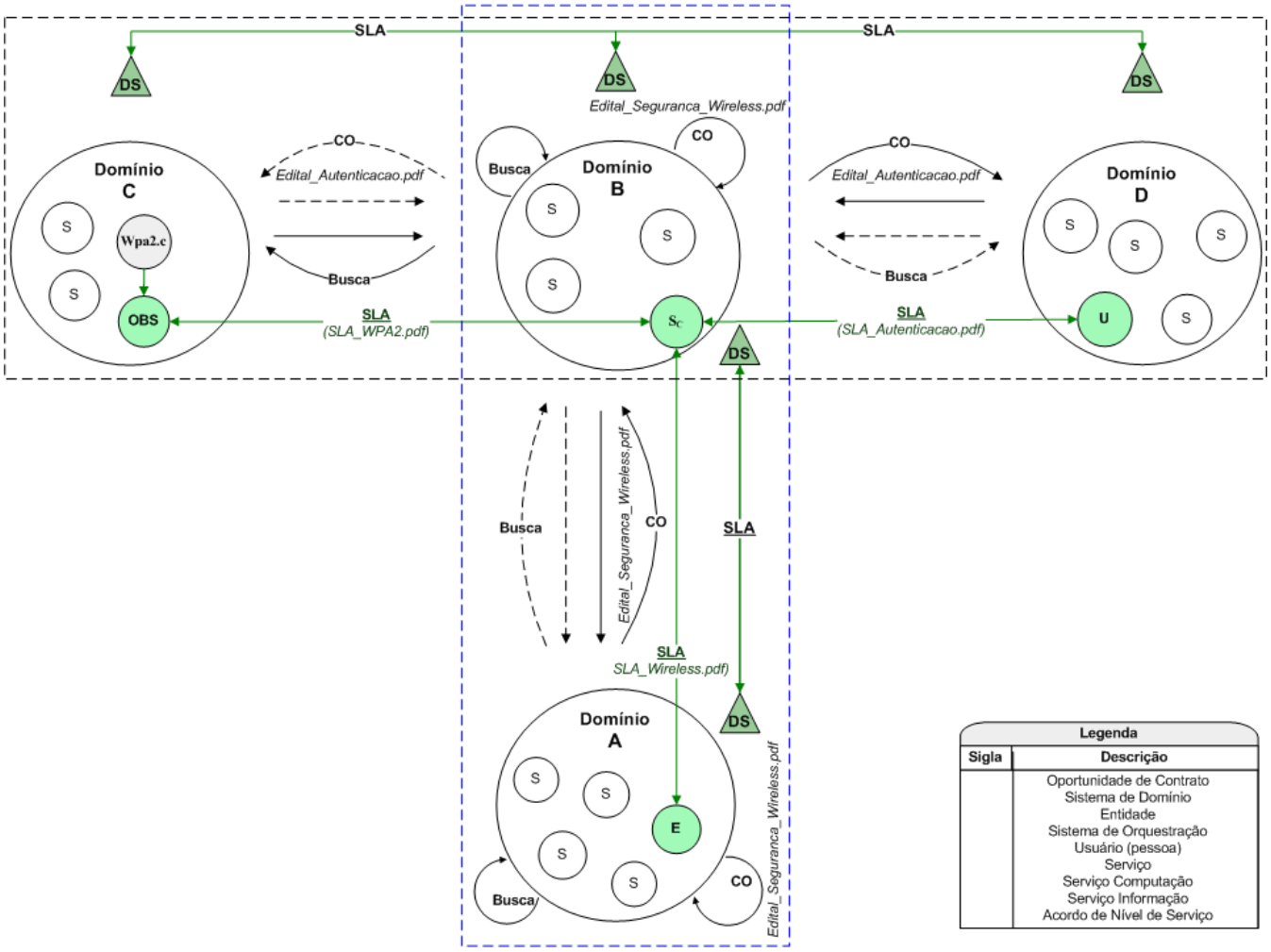

Figura 5: *Publicação de CO e contratação de serviço em múltiplos domínios.*

Na **Figura 5** 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. Então, 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.pdf*) em ambos os domínios. Esta CO descreve basicamente o interesse da entidade E em contratar um serviço de segurança para redes *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.pdf*) entre as entidades OBS e S_C, sendo o serviço compilado e disponibilizado a 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.pdf*) 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. Então, 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.pdf*) 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.pdf*) 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-se de uma arquitetura auto-similar, que independe de escala, pois a mesma solução pode ser usada em níveis diferentes da hierarquia de domínios.

4. ANÁLISE COMPARATIVA

Nesta seção são comparadas e discutidas as características mais relevantes entre as arquiteturas apresentadas. No CASCADAS o protocolo *Goal Needed* (GN) descreve semanticamente os serviços necessários em um dado momento da composição dinâmica. No XIA, a divulgação se dá por meio da escolha de nomes legíveis representativos da funcionalidade de cada serviço. Já no GS todas as entidades devem receber um nome legível em linguagem natural, possuir um ou mais objetos descritores para descrevê-las, e por fim a rede deve ser estruturada com base em ontologias publicadas.

Com relação à identificação dos serviços, somente as iniciativas XIA e GS contemplam este aspecto através dos SIDs (*Service Identifiers*) e IDs inequívocos, respectivamente. A identificação única é importante por diversos motivos, como por exemplo, rastreabilidade, melhoria da segurança e por identificar sem repúdio à autoria de comportamentos ilícitos.

No que diz respeito à descoberta de parceiros para a composição de serviços, o CASCADAS utiliza novamente o protocolo GN/GA e leva em conta o contexto exato do serviço desejado. No XIA a descoberta ocorre apenas por meio dos nomes. Um aspecto comum a estas abordagens é que elas utilizam *software* distribuído para este fim. Já o GS, utiliza uma estrutura baseada em ontologias e por meio do SDS possibilita as entidades realizarem a descoberta de serviços e conteúdos. Isto elimina a sobreposição desnecessária de funcionalidades.

Quanto à composição dinâmica dos serviços, o CASCADAS permite a agregação e diferenciação semântica dos serviços. Isto é feito de forma autonômica, sem interferência humana. Além disso, os agrupamentos podem ser fragmentados, quando, por exemplo, uma aplicação precisa ser reutilizada para outro fim distinto ou quando funcionalidades não são mais necessárias. No BIONETS a composição dinâmica autonômica também é suportada. Além disso, o BIONETS permite a seleção genética dos melhores candidatos a uma dada composição. No XIA, a composição dinâmica é possível

através do relacionamento entre SIDs. Mas, não existe suporte autonômico para isto. Já o GS, permite alcançar a composição dinâmica dos serviços de forma autonômica, através da auto-organização semântica ou de forma manual, através da invocação semântica. Na forma autonômica, além das próprias entidades de serviço estarem aptas a compor novos serviços, uma entidade auxiliar (OBS) representa entidades incapazes de se auto-representarem. Por fim, caso não seja encontrado algum serviço condizente com a demanda, esta abordagem conta ainda com a publicação de COs.

A transparência com relação à localização 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 mapeia uma coisa na outra dinamicamente. No GS cada serviço é unicamente identificado e sua localização é armazenada separadamente em um Sistema de Tabelas *Hash* Distribuídas (DHTS).

A negociação e a contratação no CASCADAS também usam o protocolo GN/GA. Um detalhe interessante é que ela é feita diretamente pelos serviços de forma distribuída, sem qualquer entidade centralizadora. Isto é fundamental para o suporte a auto-organização autonômica. A contratação usa um ambiente de comunicação orientada a conexão entre agentes do framework DIET. No XIA, não existem mecanismos implementados para a negociação e contratação de serviços. No GS o contrato é formalizado por meio de um SLA e 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. 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.

Quanto ao suporte da confiança na prestação de serviços, as abordagens CASCADAS, XIA e GS oferecem este suporte. As abordagens CASCADAS e GS incluem também um sistema de gestão de reputação de serviços. No GS existe o RS, que permite entidades (clientes) estabelecerem relações de confiança com outras entidades (provedores de serviço) e vice-versa e avaliarem o QoE e QoS dos serviços contratados.

A mobilidade é nativa no CASCADAS, XIA, BIONETS e GS. O suporte a mobilidade generalizada é fundamental em redes futuras, pois cada vez mais os usuários estão utilizado 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. Trata-se de um desafio considerável. Soluções como a proposta pelo XIA, que fazem o desacoplamento de identificadores e localizadores, tem se destacado no suporte à mobilidade generalizada.

Nos quesitos adaptabilidade e autonomicidade, o CASCADAS oferece suporte as chamadas propriedades *-*awareness*, que significam suporte a ciência de diversos aspectos (o asterisco identifica cada aspecto). 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 auto-conhecimento 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 de contexto. Ou seja, 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, auto-configuração, auto-proteção, auto-otimização, auto-contextualização,

auto-gerenciamento, etc. No GS, as entidades são proativas, cientes da situação e do contexto, capazes de aprende com suas experiências e tomar suas próprias decisões com o mínimo de interferência humana.

No BIONETS os nós T fazem 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 possui ainda 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. Ou seja, não existe mecanismo explícito que permita que os serviços se adaptem à 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.

Quanto ao requisito da evolução, o CASCADAS, o BIONETS e o GS oferecem evolução autonômica. Ou seja, 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 o requisito de interoperabilidade com as redes IP. No BIONETS é possível estabelecer a comunicação entre os ambientes BIONETS e IP utilizando um AP, que conta com a presença de um servidor *Proxy* capaz de traduzir operações BIONETS para as redes IP e vice-versa. O XIA utiliza o *eXpressive Internet Protocol* (XIP), protocolo de camada de rede baseado no IP e proposto para substituí-lo. Já no GS, todos os sistemas "vivem" em um ambiente virtual, podendo ser aplicado para ambientes IP, não IP e pós IP.

A Tabela 2 sintetiza um comparativo entre as abordagens relacionadas.

Princípios	**CASCADAS**	**BIONETS**	**XIA**	**GS**
Descrição	GN abrange descrição 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		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, autonômica centralizada e autonômica distribuída.
Localização transparente	Desacoplamento via publicação/assinatura de eventos.		Desacoplamento via servidor de *lookup*.	
Negociação	Direta usando GN/GA		Não possui	Usando o PSS.
Contratação	Direta usando agentes DIET		Não possui	SLA como parte integrante da arquitetura.
Gerenciamento	Supervisão pervasiva dos serviços		Não possui	
Confiança	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*.	Oportunidades de Contrato, Sistema de Reputação, OBS
Autonomicidade	Amplo suporte de funcionalidades auto-*, tais como auto-organização, auto-otimização, auto-proteção	Auto-geração de serviços (autocatálise)	Manual	Suporte a funcionalidades auto-*
Evolução	Autonômica, via agregação/diferenciação, 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. Via agregação/diferenciação, adequação e reusabilidade de serviços.
Interoperabilidade com Redes IP		IP e não IP.	IP	IP, não IP e pós IP.

Tabela 2: Comparação de Abordagens.

5. CONCLUSÃO

Neste artigo, foram apresentadas algumas abordagens relacionadas à ambientes orientados a serviços, bem como uma proposta de uma nova arquitetura de redes centradas em serviços.

A arquitetura proposta aborda questões fundamentais para uma nova 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 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.

REFERÊNCIAS

CARDOSO, J.; WINKLER, M.; VOIGT, K.; BERTHOLD, H. 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, M. P.; TRAVERSO, P.; DUSTDAR, S.; LEYMANN, F. Service-Oriented Computing Research Roadmap. 2006.

CHANNABASAVAIAH, K.; HOLLEY, K.; TUGGLE, E. M. Migrating to a Service-Oriented Architecture. IBM Software Group. April 2004.

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

LIU, S.; BI, J.; WANG, Y. A DHTs-Based Mapping System for Identifier and Locator. 2009

CIRANI, S.; VELTRI, L. Implementation of a framework for a DHT-based Distributed Location Service. Dpt. Information Engineering, University of Parma, Parma, Italy. 2007.

MARTINS, B. M. Desacoplamento de Identificadores e Localizadores: Uma Comparação de Abordagens e Proposta de Arquitetura. 2011.

JACOBSON. V.; SMETTERS, D. K.; THORNTON, J. D.; PLASS, M. F.; BRIGGS, N. H.; BRAYNARD, R. L. Networking Named Content. Palo Alto Research Center (PARC). ACM CoNEXT. Rome, Italy. December 2009.

MANZALINI, A.; MANNELLA, A.; MOISO, C.; HASELOFF, S.; KUSBER, R.; BRGULJA, N.; ZAMBONELLI, F.; DEUSSEN, P. H.; SAFFRE, F.; BARESI, L.; LENT, R.; GELENBE, E.; FERDINANDO, A. D.; CASCELLA, R. CASCADAS - Bringing Autonomic Services to Life. Deliverable 8.4. January 2009.

HOFIG, H.; BENKŐ, B. K.; DI, N. E.; MAMEI, M.; MANNELLA, A.; 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; XIRUHAKI, C. Agents in Decentrailised Information Ecosystems: the DIET Approach. In Proc. of the Artificial Intelligence and Simulation Behaviour Convention (AISB 2001). York, March 2001.

BAUMGARTEN, M.; ZAMBONELLI, F.; CASTELLI, G.; BICOCCHI, N.; KUSBER, R.; BRGULJA, N. Final Report on Open Toolkit for Knowledge Networks and on Advanced Knowledge Networks Concepts and Models. Deliverable 5.4. December 2008.

LINNER, D.; PELLEGRINI, F. D.; MIORANDI, D.; MOISO, C.; BACSARDI, L.. BIONETS Architecture: from Networks to SerWorks. IEEE. 2007.

TSCHUDIN, C; 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, D.; YAMAMOTO, L.; DINI, P. Service evolution in bio-inspired communication systems. European Commision. Project: BIONETS. 2006.

ANAND, A.; DOGAR, F.; HAN, D.; LI, B.; LIM, H.; MACHADO, M.; WU, W.; AKELLA, A.; ANDERSEN, M.; BYERS, J.; SESHAN, S.; STEENKISTE, P. XIA: An Architecture for an Evolvable and Trustworthy Internet. January 2011.

PELLISSARI, F. R.; RIGHI, R.; WESTPHALL, C. 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.