Wireless Pers Commun (2011) 57:393–411
DOI 10.1007/s11277-010-0076-7

Fifth Generation Networking Principles for a Service Driven Future Internet Architecture

Ram Kumar

Published online: 15 July 2010

Abstract The vision of all-IP networks where IP forms the simple common layer understandable across the whole network has undeniable advantages. However, such simplicity comes as a major hurdle to flexibility and functionality to the architecture. This is evident from the increasingly numerous and complex engineering solutions and optimizations required to accommodate essential qualities like mobility, security, realtime communication support etc or to mitigate the shortcomings inherent in the 'traditional Internet' architecture. While a clean slate approach to address these shortcomings is not an option in a realistic scenario, it is important to examine the architecture as a whole to address emerging network requirements and overcome existing shortcomings at the architecture level rather than engineering solutions to an existing inefficient one. This architectural re-examination should also facilitate discussion into what design principles for future generations of Network Architectures which will eventually replace the design tenets for the current Internet. While 3G and 4G systems were more focussed on convergence towards an All-IP network and some improvements in the core network, the architectural design remains stagnant with layered paradigms and inherent inefficiencies. A departure from this shackled approach could be the distinguishing feature of 5G systems and beyond. We claim that there is a pressing need to move towards a Next Generation Network architecture built to natively support requirements such as network resource abstraction, mobility, security, enhanced routing, privacy, context communications, QoS, parallel processing, heterogeneous networking etc. Instead of treating the network as just providing connectivity specified by endpoints, it is of great advantage to applications to recognise it as a *service* characterized by attributes, abstracted to a higher level to represent a collection of capabilities that the network offers. This uniform high level abstraction can effectively mask the heterogeneity and implementation discrepancies in the underlying infrastructure. Besides, in a network environment where an connectivity instance might transverse diverse business/ownership/capability domains, the approach proposed in this article can provide a transparent abstraction for resource negotiations across the domain to be available for end-to-end setup. This architectural change should also be manifested

R. Kumar (✉)
Faculty of Science and Engineering, University of Agder, Grimstad, Norway
e-mail: ram.kumar@uia.no

according to the principles of SOA to ensure interoperability, backwards compatibility and migration. In this article, we introduce a Service Oriented framework and network architecture aimed at tackling the heterogeneity of emerging requirements and proposed solutions into a coherent interoperable architecture using Web Services specifications as the basic standards. We propose to model the new architecture on relationships between entities and discuss the motivation this new architecture in the form of a new framework called ROSA.

Keywords Future Internet · Network architecture · Service oriented architecture · ROSA

1 Introduction

The Internet, as a network of connected computers, came into existence in the 1970s [1]. The early Internet interconnected a stationary set of nodes. The architecture and the protocols used were designed to accommodate and exploit this simple stationary nature. The designers wanted to build a network infrastructure to interconnect all computers in the world together and provide a framework for yet unknown applications to be invented and run [2]. The nodes were well described by IP addresses that identified the nodes directly on the network. Routing protocols then took advantage of the static nature of the Internet.

Although the usage and possibilities of the Internet have expanded beyond its initial scope, the design principles and architecture of the original Internet are still followed today. The success of the Internet has by itself shackled the possibility of any dramatic change or a completely new architecture from being implemented to accommodate the requirements that were not envisioned in initial design stage. The massive installed base of routers, clients and other network equipment supporting today's network infrastructure makes sure that any significant changes to the architecture of the Internet protocol (IP) based network will be overlooked, if not ignored. The financial aspects of migration will play an important part in migrating to any other replacement proposed from today's architecture.

Due to the explosive evolution of the Internet into what it is today, it is difficult to clearly define 'the scope of the Internet' or specify 'edges of the Internet'. The boundaries of the legacy internet (if we can call the earlier iterations of the Internet that) is continuously being blurred by the introduction of new devices, violation of the original design tenets and the incorporation of new paradigms. Currently, a typical view of the Internet implies *an electronic communications network that connects computer networks and organizational computer facilities around the world*. This is a very generic definition which makes the term 'Future Internet' a very wide area for research and a 'Future Internet Architecture' a generic network architecture that could address all networks.

2 The Need for a New Architecture

With time and technological advances, the networking solutions have been steadily increasing in complexity. To accommodate new requirements, the Internet has been engineered with more powerful routers, faster backbones, faster processing at end points and traffic shaping to accommodate new models and better performance. These often incompatible engineering solutions (or 'hacks to the original stack') provided temporary fixes to new problems but introduced unnecessary complexity, a few of which will be singled out in the later sections. There exist fundamental inefficiencies in the current network architecture which cannot be

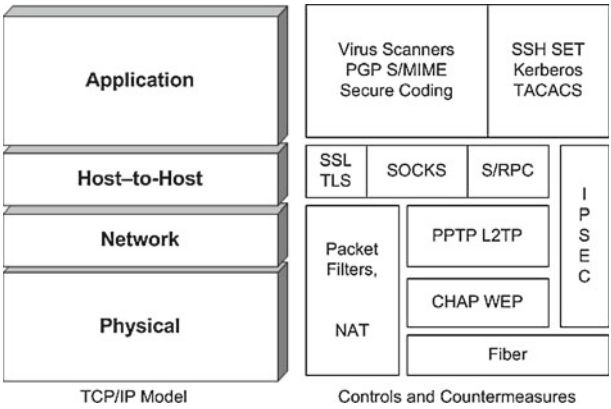

Fig. 1 Internet security controls and countermeasure [6]

addressed efficiently. For example, challenges with mobility, end-to-end Quality of Service (QoS), security, trust etc. In implementing the work-arounds, the Internet today breaks most of the design principles initially conceived for it [3]. Some examples can be perceived from 'end to end' principle violated by middle boxes, NATs etc. [4]; fairness restricted via traffic shaping, packet inspection; best effort breached with overlays; stateless network concept infringed by intelligent middle boxes [5], stateful proxies, label based router etc. The vastness and distributed control nature of Internet today, makes it difficult to implement distributed applications with realtime guarantees necessary for certain types of communications. These changes do not converge towards any new architectures, but are add-ons or overlays piggybacking on the same old design. Some of the urgent problems like address exhaustion, better compatibility to emerging technologies via header extensions and better security are squeezed into the IPv6; which still addresses only a part of problem, not the network architecture limitations as a whole.

We can find an instance of this complexity in the current Internet security architecture. Security was not as important a concern as openness and fairness during the birth of the ARPAnet. The fact that the network placed no restrictions on connectivity meant that innovative applications could be deployed without obstacles, which essentially lead to the growth of the Internet to the magnitude we witness today. However, the very same design tenet has now made protecting the network from malicious hosts very difficult. For example, while rudimentary security measures solve most of the problems (e.g., security holes in an applications can be patched and end-to-end security protocols can be deployed, or security overlays for specific protocols), the openness has made it difficult to defend against Denial of Service (DoS) attacks. A visualization (Fig. 1) adapted from [6] indicate the complicated and patched security architecture of the current Internet. The lack of a harmonized security strategy and multiple approaches manifest themselves as cross layering and conflicting overlays. The lack of a common trust, privacy and security approach is just one of the shortcomings of the current Internet architecture. The Ambient Networks project, a European sixth frame work project identified some of the requirements to be addressed for their next generation communication architecture [7]. Haggle [8] identifies that the root cause for some of the usability deficiencies with regards to mobile devices today arises from the synchronous IP-based APIs presented to applications along with the numeric addresses as end-points. Applications implemented in such models rely on networking infrastructure for end-to-end

communication without taking advantage or being aware of local or neighboring network resources. We list a few of the challenges or requirements encountered by applications below:

- Concept of Location/Neighbourhood awareness, proximity etc.
- End to end service oriented communication.
- Separation of identifier and location in naming and addressing.
- Session continuity and management across domains.
- Common trust, anonymity and federated identity management.
- Parameter/Metric based routing (added value based routing).
- Routing facilities based on application layer needs.
- Efficient Mobility, Multihoming, delegation, indirection.
- Capability signalling across devices, domains.
- Real-time and Distributed real-time application requirements like priority, guarantees etc.
- Scalability for trillions of nodes.

As can be observed from the above list, the issues to be addressed are rather basic and spread across the existing 'layers' of TCP/IP model; i.e., it is difficult to solve the above shortcomings at a particular 'layer' of the current Internet architecture. But, these arguments in no way propone demise of the Internet, but suggest the strain impressed on the legacy architecture it still follows. The stress on the current architecture is not limited to the new usages of existing technologies, but also arises from new technologies that are incompatible, but forced into compatibility for legacy interoperability. For instance, ad hoc, vehicular and sensor networks differ dramatically from the relatively static 'client-gateway-server' design of the Internet with the number of nodes stretching into billions and extremely dynamic mobility scenarios.

While the internet has tried to avoid vertical silo (smoke stack/chimney model) effect by abstracting horizontally with layers rather than complete end to end solutions, in reality, the effect has been more hard-coupled. The proliferation of IP as the de facto standard for network abstraction has made the generic concept into an IP-Hourglass model [9]. This model has worked remarkably well over the past few decades, but does not perform well when exercised against new paradigms and applications. There is an interesting discussion growing around the 'waistline' of the internet with various opinions and concepts emerging on 'what will or should' be replace or added to expand the waistline of the current network architecture [9,10]. With the shortcomings of IP (such as the identifier/locator dichotomy), it is only logical to provide an alternate addressing scheme to take advantage of the innovation in the networking and routing domains over the past few years. However, *what is the best alternative?* is still an open question. It is this openness that should be embraced rather than providing a solution which in a few years will find itself inefficient or even unsuitable for use due to newer unforseen requirements. In a best effort packet delivery mechanism like IP, the concept of a separate control channel is diminished (apart from already existing internal and external routing protocols) [9]. Each packet carried enough information for processing at intermediary nodes. Following convergence and adoption of IP as the de facto abstraction at the network layer, the concept of the self contained packet became the norm. This simple and common layer was advantageous for interoperability but is manifesting as a major challenge for the flexibility for applications running at higher layers. With the new requirements like QoS and security, extra intelligence needed to be built into routers to examine the contents of each packets to decipher what needed to be done with it for additional functionality (as per end user signalling through RSVP, for example). With the emergence of realtime multimedia as a major part of the traffic on the internet, separate control protocols had to be developed (SIP, RTP) just to facilitate the delivery of multimedia streams/sessions. With QoS, routing is no longer simple forwarding of packets but consists of prioritizing, queuing, dropping,

tagging/marking and so on. The routing architecture of the current Internet does not support packet forwarding based on such rich or descriptive parameters.

These trends segment the Internet as a collection of application level networks (torrent, IMS etc), each overlay addressing a specific application requirement or functionality. Different control protocols implemented in a distributed fashion decide the nature of the overlays. As articulated by Aguiar [9], the concept of in-band control signalling through the packets necessitates the 'hard' processing of each packet to provide additional functionality to the flow. This is hardly efficient when the choices become numerous and the packet count follows suit. A separate control plane or architecture might be necessary, independent of data flow to facilitate added functionality to the communication via networks. This will facilitate capability negotiation across different control domains (like businesses, Autonomous Systems etc.) for setting up sessions or temporary peering agreements separate from data delivery mechanisms. The proposal to have an additional hourglass model for the control architecture complementing the data hourglass model (with IP at the waist) [9] for networking might be one approach to address this challenge. The idea of separation of concerns (control and data) within the network architecture brings flexibility at the cost of simplicity.

2.1 New Paradigms

The Internet evolution has been characterized by ingenuity on the part of software and application designers to circumvent the architectural limitations, and has brought into play varying 'players' into the sphere of influence [11]. The resulting 'tussle' influences not only the direction of the future evolution of the Internet, but also the nature of the next generation architecture like notions on design, space for tussle etc.

Access technologies and applications emerge independent of the Internet, which in due course requires interacting with the network for various reasons. We find powerful mobile communications devices, sensors, medical implants, vehicles and a host of other network capable appliances emerging. Besides, new networking models like ad-hoc networks, vehicular and sensor networks present challenges which are not efficiently handled by the current architecture. The number of connected nodes in the Internet has gone from a few in the early eighties to millions (currently), with a strong possibility to be trillions [12] with the inclusion of ad-hoc nodes, cheap sensors and networked vehicles in the future.

User demands like interactive resources, user generated content, content sharing, local access of distributed content, anywhere/anytime access of own data etc. requires the underlying architecture to support a set of basic capabilities, which are inefficiently implemented into the legacy Internet [13]. The demands can also include demands based on or on behalf of the users by other entities such as governments, corporations and content owners. For example, the open and end-to-end nature of the Internet is broken by middle boxes to accommodate for address exhaustion, security etc. [4]. This necessitates architectural changes incorporating such additions.

These are not independent driving forces. These factors tend to influence each other to a stage where the underlying architecture can no longer efficiently support the newly construed paradigms. This will be the case with any new tightly designed architecture. The boundaries of such architectures will be tested. One of the more accommodating architecture would be the one which account for this growth ('design for tussle' [11]), or a system modular enough to keep pace with the innovations around it. The idea of service oriented architecture and service composition, manifested in different proposals as network composition is worth considering in this context.

3 Challenges to a New Architecture

There are two common methods of approach that can be utilized when thinking forward. One is the *Incremental Approach* aimed at maintaining backwards compatibility while migrating towards a new architecture (E.g. IPv4 to IPv6 migration). The other approach is to have a *Clean Slate Thinking* to fix all the problems that can be identified as being inherent in the current architecture. Architectural changes to the core of the Internet (E.g.: IPv6) and add-on/overlay services (E.g.: MIPv4, MIPv6, IMS etc.) have met varying levels of success. There are a host of new architecturally superior implementations and changes dismissed a priori by the marketplace, due to reasons such as ossification of the TCP/IP model and business aspects of bringing about a dramatic change in the currently installed infrastructure base. It is easier for researchers to consider a clean slate approach of a new architecture, protocols and service. However, such an approach is generally unacceptable due the changes required to already existing infrastructure and devices and due to the disruption of existing businesses. This approach, while ideal from a research standpoint is difficult to implement in the current scenario.

The incremental approach to addressing the current limitations is attractive to service providers and network operators in terms of cost and availability. This approach often produces inefficient and often complicated solutions. An ideal solution to this conundrum will be to suggest modular incremental changes to current architecture aimed at addressing immediate problems, which function as milestones or part of the transition towards a completely reconsidered and modular network architecture. The idea of overlay networks (for example, Peer-to-peer overlay networks [14]) is quite relevant in this context, where a new technology can be implemented at 'present time' over existing architecture, so as to bring in the new functionality without a radical change to the underlying architecture. This functionality, at a later time can be accommodated as a part of the architecture itself, if designed to relevant open standards.

The above mentioned approach is not only true for functionality overlays like IMS, but completely reworked architectures as well. Consider the migration towards IPv6 from IPv4 in this context. The newer Internet layer (IPv6) can coexist with the current one (IPv4) via gateways connecting islands of IPv6 routers to the IPv4 world, software encapsulation like the 6to4 transition mechanism [5] or tunneling etc. [15]. In the future, when most of the nodes (routers, specifically in this case) support IPv6 in the future, the IPv6 'islands' automatically become the 'main network', with IPv4 becoming 'legacy islands' interfaced via 'legacy' gateways. This approach however requires a scale, consensus and collaboration from major players in the research community and industry. A study of the current state of the art will reveal that both approaches are being explored by various entities globally, substantiating the necessity and urgency of such a transition [10,16].

4 Existing Approaches

Under the current network architecture, the applications are responsible for setting up all bindings required for communication. This necessitates that the software is written to specific underlying network architecture, without modularity. The close binding also makes it difficult for developers to implement applications and solutions that can adapt to new communication mechanisms. Most of the new 'flexible generic platform' proposed to overcome such limitations concentrate on abstracting the applications from the underlying network architecture, mostly by adding one more abstraction layer over the existing naming system

(IP). These approaches, to an extend, mitigate the ill effects from current dual usage of IP addresses as end point identifier (name) and location (address). The reliance of certain applications on the Domain Naming System (email, web addresses etc.) together with the inability of the DNS to adapt to rapid updates make it more difficult in dynamic mobile environments. The migration from IPv4 to IPv6 will provide some temporary relief to pressing issues such as address exhaustion, resource allocation support via header extensions and improved security features. But, IPv6 does not address the architectural limitations of the Internet. There are various proposal and projects at different stages of maturity being considered by the Internet community, industry and academia related to the architectural nature of the Future Internet.

Host identity protocol [17] identifies the naming and addressing of entities as the key challenge in today's architecture and proposes as a solution to separate them, decoupling the usage of the address (i.e. the IP address) as the identity of resources or nodes. The separation is achieved by introducing a new layer between the conventional TCP/IP stack between the network layer and the transport layer. HIP uses cryptographic identifiers as the Namespace which helps to integrate baseline end-to-end security into the architecture when used with Diffie-Hellman [18] and appropriate security protocol, such as Encapsulated Security Payload (ESP) [19]. Each node has a private/public key pair and the node's identity is a hash of its public key. Several solutions have been proposed to accommodate mobility and Multi-homing [20,21] using HIP. However, there are some undesirable consequences where the node loses its identity if the public key is ever changed or compromised. Haggle approaches these challenges using the concept of 'Pocket Switched Networking' [22,23] to take advantage of both infrastructure based and Ad Hoc (peer to peer) communications opportunistically. The Haggle network architecture is aimed at providing seamless network connectivity and application functionality in mobile environments by separating application logic from underlying network architecture.

Ambient Networks (AN) [7,24] introduces the concept of horizontally structured mobile systems that offer common control functions to a wide range of different applications and interface technologies to provide a common networking concept to adapt to varying heterogeneous wireless and service environments. The AN naming architecture adopts a layered naming model, with separation concepts borrowed from layered naming architecture [16] and HIP [17]. Dynamic bindings at different layers enable the basic mobility of nodes, 'bearers' and applications [25].

Data-Oriented Network Architecture (DONA) [26], borrowing heavily from other exercises like TRIAD, SFS and HIP, suggests a clean-slate redesign of Internet naming and name resolution, to address specific features such as persistence, availability, and authentication for service access or data retrieval. DONA justifies this proposal pointing out the existing discordance between historical design (host-oriented) and current usage (data-oriented). However, the clean slate approach is mostly limited to how Internet names are structured and resolved. As with HIP, DONA replaces DNS names with flat, self-certifying names, and replaces DNS name resolution with a name-based anycast primitive that lives above the IP layer. DONA proposes that names handle persistence and authenticity, while name resolution handles availability. There are other attempts to approach the problem from enterprise perspective through Application Oriented Network Architecture (AON) [27], borrowing ideas from NGN architecture [28].

Another approach is to extend today's architecture using middle-boxes (TRIAD) to avoid the migration to IPv6, or base the architecture on a new central parameter or paradigm. DONA and Haggle architectures revolves around 'data' as the center piece. There are advantages as well as pitfalls to all these approaches. For example, when the architecture is designed

around data as the most important parameter, then the design must encapsulate and manage all data as haggle proposes to do. This is not an ideal approach as other applications might not want to relegate ownership of their data to middleware. Identifier-Locator Network Protocol (ILNP) [29] proposes an extension of the 8+8 for IPv6 [30] idea and SHIM6 [31]. ILNP specifically addresses mobility and multihoming issues with current implementations, integration of middleboxes like NAT, enhancement of end-to-end security, among other features. Reliance on DNS for location update of mobile nodes apart from ICMP locator updates might be a shortcomings when dealing with highly dynamic and unreliable environments.

There is a significant amount of discussion forming around the EIFFEL project [32] regarding the nature and process of arriving at the Future Internet, albeit from a European perspective. This project, not limited to technological issues but also social issues surrounding a future networked society, is organized around a number of Technical Areas (TAs) focusing on technological, societal, regulatory and policy-related questions. For a project of such magnitude, it will be a while before its future internet architecture proposal emerges, to be studied or compared to existing solutions. The projects mentioned here are not a conclusive list of attempts to address the architectural shortcomings of the Internet. They are too numerous and outside the scope of this article. But, the conclusion derived implies that the research community and industry recognizes the shortcomings and are exploring various approaches to address them.

5 A Different Approach to Networking

We develop our vision from a top down approach, from the point of view of the application developers. From such a perspective, networking is not just connectivity specified by an end point tuple (as in Berkeley APIs). A network is a collection of distributed *services* that are *available* to the applications. The application developer should not worry about the state of the network at application runtime during the application design. This decoupling of addressing network end points directly in applications can be achieved through a uniform approach to exploit the underlying network infrastructure via a generic, rich and standardized interface. Such an interface provides abstraction of network capabilities to applications and decouples the heterogeneity arising from the below mentioned factors. Adapting to heterogeneity forms one of the basic characteristics of our approach. Heterogeneity in various layers of the current architecture arise from various sources including:

- Network technologies, devices and Operating Systems
- Middleware solutions and communication paradigms
- Programming models and languages
- Services and interface technologies
- Domains and architectures
- Data and document formats
- Nonfunctional aspects such as information models, security, availability, transactions etc.
- Business borders
- Communication procedures and security policies.

5.1 Network as a Service

We claim that the layered paradigm in networking architecture is not the ideal approach for abstraction. As with Haggle, we propose a layer-less architecture which abstracts the

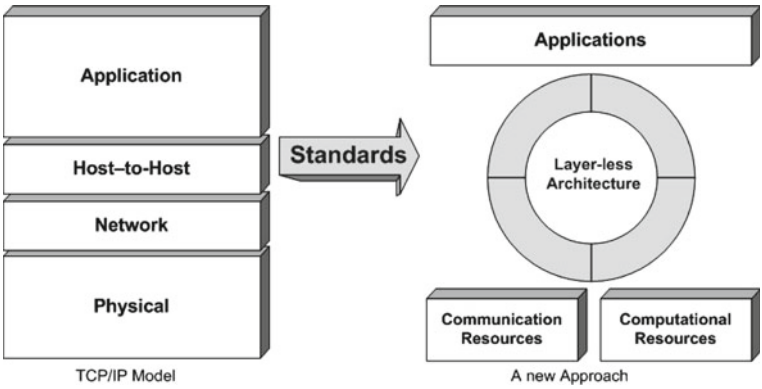

Fig. 2 The vision of *network as a service*

underlying connectivity and network computational resources to applications via a high level API. Applications should not be burdened with attaching themselves directly with end points or the connectivity status of various available interfaces; it should automatically be taken care of by the underlying architecture. We argue that ideal network connectivity is a service that any application should be able to use. It is a service that is composed from underlying (possibly orthogonal) capabilities of the node and the environment that the application then resides in. This obvious statement enables us to look at architecture from a service oriented point to view (Fig. 2). This high level abstraction for a network has various advantages. Such and approach inherently accommodates the business boundaries existing in the real world networks such as commercial boundaries, administrative domains etc and helps traversing across these a natural part of service negotiation and usage. While this is also the approach of overlay networks, their implementation is not uniform or generic enough to apply to the network architecture as a whole. We propose to integrate the available communication services and current higher layer services in a unified and standardised manner.

5.2 Network Service as a Collection of Services

The next step is to identify the basis with which we compartmentalize the capabilities into modules, which can be later orchestrated. We take cues from the 'tussle' [11] being played out in the networking world to impose boundaries on the modules. A module encompasses a well defined service small enough to be a factor in the tussle but large enough to provide a specific, well defined, usable and non-trivial service. Identifying a set of modules which provides the services needed to compose a network architecture is not straight forward considering the fact that as a 'future' network architecture, it should be able to gracefully accommodate a wide spectrum of potential uses. The borders of modules and which attributes to consider as a module is of course, an open question. Different approaches can propose different modules to accomplish the same goals and it might be probable that it is impossible to agree on a standard set. But, as long as different modules provide a uniform interface and a clear description of its capabilities, the architectural principles we propose hold true.

From the above research directions, a sense of future direction can be derived. We can see not one 'Internet' but many 'Internets' of varying capabilities and characteristics (inter-network of things, inter-network of specific applications, inter-network of specific intentions etc.). This concept is already starting to evolve if we consider Peer-to-peer application 'networks' using the existing infrastructure as a transport network. This brings us very close to a similar problem that existed in enterprises towards the beginning of the twenty-first century. Enterprises built their IT systems to streamline their processes. Large distributed enterprises built middleware to support transactions and interconnect their systems across domains. Since there were no standards for such systems, these proprietary systems posed a problem during instances of interoperability (mergers, acquisitions, collaborations etc). The concept of Service Oriented Architecture (SOA) [33] was adopted to enable a standardised and open way for enterprises to open up their IT infrastructure for collaboration. We note the best known implementation of SOA in Web Services (WS-*) [34] specifications, standardization and its implementations. While, Web Services specifications specifically address Enterprise Application Integration (EAI [35]), we need a similar principle and framework to be applied to future heterogenous communication networks in order to interconnect business borders and administrative domains.

The simple sounding business goal of *connecting to customers, suppliers or partners electronically* [36] translated into web services that offer standard based mechanisms to create or compose services from composite and often cross-organizational components and Web based services [37]. We look at the underlying network architecture under the same requirement considerations, i.e, a service to be offered to applications, composed of various other services, local or distributed.

5.3 The Principle of Service Orientation

Before we apply service orientation principles to the network architecture, it is helpful to identity the definition of a *service*. A service as specific functionality is characterised by the following features [38]:

Composable: An entity can use the service depending on the conditions specified either directly or as a part of another service.
Describable: A service can be fully described including what functions it provides and how these functions can be accessed (e.g. through metadata).
Discoverable: A service can be discovered based on their metadata by other services or entities.

SOA represents an abstract architectural concept of building software systems that is based on loosely coupled components (services) that have been described in a uniform way and can be discovered and composed [38]. The services that form the part of an SOA should be dynamically composable by any entity interested in availing it. The core elements that comprise an SOA is illustrated in Fig. 3a. This architecture is further extended with a Service Bus (SB) to make the discovery and binding process more transparent to the requesting service by visualizing the candidate services from the requestor's perspective (Fig. 3b). The concept of SB is important as it forms a decision making entity or a middleware on behalf of the requesting service.

We adopt this updated SOA approach to construct the elements for a Network Architecture. The abstraction of network as a service or a collection of services can be gracefully accommodated into this model.

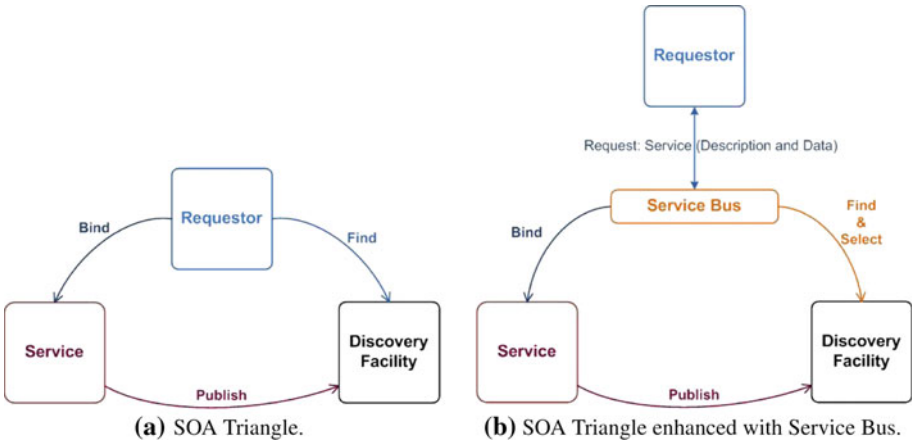

Fig. 3 Basic components of SOA architecture

6 A Relationship Based Service Oriented Architecture

To propose a network architecture within the structure of SOA, it is necessary to define a framework or a collection of services and define the environment within which it functions. This is not a straight forward exercise, given that our intention is to define a generic architecture for the next generation communication networks. To arrive at such a architecture, we inspect the highest level abstraction required for any communication. When two entities meet (have access to a channel with another entity) and wish to communicate with each other. The *entity* here do not particularly refer to a node, a device, a piece of software or a service but could be any of the them. We would like to be agnostic here regarding where or what implements the intelligence for communication. We visualize the wish to communicate of two entities (with the intent and ability to do so) as the meeting of two *domains*.

Domains can be conceived as a region (physical or virtual) characterized by a specific feature and restricted by boundaries. A generic example for such a domain could be a business with business boundaries and characterised by business processes. This gives us a generic enough platform where services can reside and interact. For example, a transaction between services of different business entities (B2B) naturally classifies into a meeting of business domains. A physical device forms a domain with the common characteristic of a physical boundary and (usually) a single ownership. In this case as well, connecting two devices together can be reduced to a meeting of two domains. This *Concept of Domains* can be extended recursively for already collaborating domains, i.e, when two domains meet and agree upon the various aspects of service sharing and usage, their collaboration can be again abstracted as a domain, which being the collection of capabilities that they together can perform. The implications of such recursive nature derives from the composability of services within an SOA.

In this paper, We define 'Relation' as an association among dynamically collaborating nodes, devices and services in a network, characterized by a 'relationship metrics'. We propose a frame work termed 'Relationship Oriented Service Architecture' (ROSA) [39] to agree upon a broad vocabulary that will be used to model recurring themes in ICT and integration environments. The aim of such a framework is to be able to reference one or more open specifications or standards for each identified service, that can be used to implement

various version of the service. This is a flexibility available in the SOA. SOA principles are considered in all aspects of design including interfaces between modules and the relationship description. This enables us to decouple certain aspects or modules and develop it independently. Besides, this also implies that the services provided by some of the modules can be accomplished by a web based service or remote entity. This would be helpful in abstracting the network architecture across domains and networks without too much reworking of the communication network. This should also simplify integration with existing infrastructure and third party service providers (like standard based trust and security service providers).

The overall architecture provides an 'Intelligent Middleware' managing the communication, while providing an API towards the applications and managing connectivity resources independently. In SOA, the actual services are usually relatively simple 'black boxes' that can be applied in a flexible fashion in a variety of instances, as such avoiding duplication of functionality. While it is not efficient to dictate that all applications orchestrate and construct their own solutions from the above services, any large scale application can do so via the APIs that these services expose. For normal scenarios, the separation of application logic from transport logic to make applications communication agnostic can achieved by providing a higher level aggregate of these services via more specific generic APIs. This is done via the 'Relationship Manager' (RM). We focus on the argument that the given modules can be composed into a coherent architecture via a single paradigm, namely Relationships. We define 'relation' as an association among dynamically collaborating nodes, devices and services in a network. A relationship description contains parameters ('relationship metrics') to express the nature and background of the collaboration.

6.1 Relationship Manager

The relationship manager is itself a service which composes available services and offers APIs to accomplish most of the common services that applications need. The RM (a simplified version of 'Enterprise Bus' in Web Services) fits the numerous service components into a logical process (Orchestration) and facilitates the translation of data flow between services that may interpret a term differently. RM helps to avoid the 'monolithic silos' traditionally formed when implementing a complete usable service. The RM orchestrates the services for processes (on behalf of applications), based on contexts. Entities always communicate with services in their environment according to a certain context. Orchestrating these services *in context* provides the definition of relationships and causalities between different services of an entity communication space [40]. The context is expressed in the context service and the RM contains the logic to use it for orchestration.

By being a service by itself, the RM is replaceable with other implementations offering similar interfaces. However, to be truly modular and avoid tussle in the core, the relationship manager should be minimal, analogous to a micro-kernel in an operating system. This means that all useful services should be implemented outside the RM, or in services space and the only service that the RM provides is a meaningful orchestration. However, even for environments where the applications directly manage the orchestration, RM must be present for bootstrapping purposes.

From a different perspective, the RM virtualizes the available services to the applications. This is similar to the concept in Fig. 3b where the Service Bus virtualizes the candidate services from the requestor's point of view. The RM can be thought of as a local instantiation of a Service Bus, which simplifies the search, query, binding and access of other services within the ROSA framework.

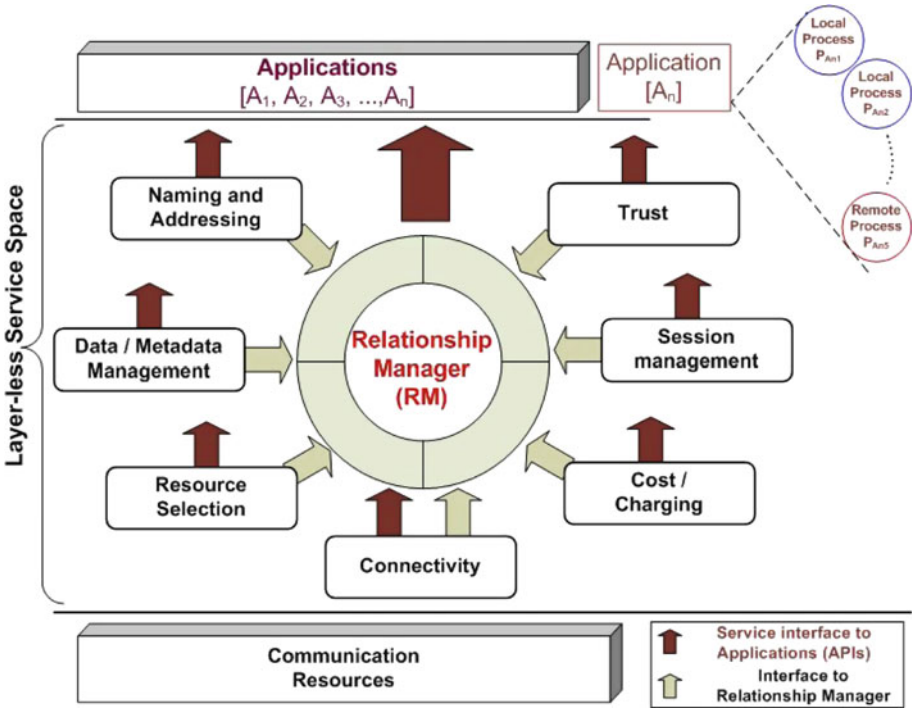

Fig. 4 A sample ROSA instance

6.2 Service Composition

Figure 4 indicates an example of how a network can be composed out of available services. The services indicated are highly aggregated services from other elementary services available. The simple file transfer scenario mentioned above can be addressed by encapsulating the available network interfaces (Wi-Fi, bluetooth, Wireless USB, NFC etc.) and related protocols via the 'connectivity' service. This approach detaches the applications from having to be aware of the device location, neighboring nodes and their capabilities. If both devices support such a framework, then the underlying services can be combined by the relationship manager to accomplish a request similar to 'transfer the file to the closest neighbor after verification, via the lowest cost path'. The application need not know whether the interface used is Wi-Fi or bluetooth or NFC, or even how the phone discovered its neighbor. The detachment of network services also enable applications to fully explore the 'late binding' (addressing the entity but postponing the entity location till the last possible instance) or 'relationship composition' by the relationship manager, as it can wait as late as possible to compose the architecture, utilizing the latest context/network/location updates and status.

6.3 Services

A selection of 'building block' services, which can be orchestrated to form higher level services is depicted in Fig. 5 to visualize the framework. The services are clustered into logical groups to aid readability and no dependencies or explicit associations exist between service definitions. In practice, if several services with similar capabilities are exposed in

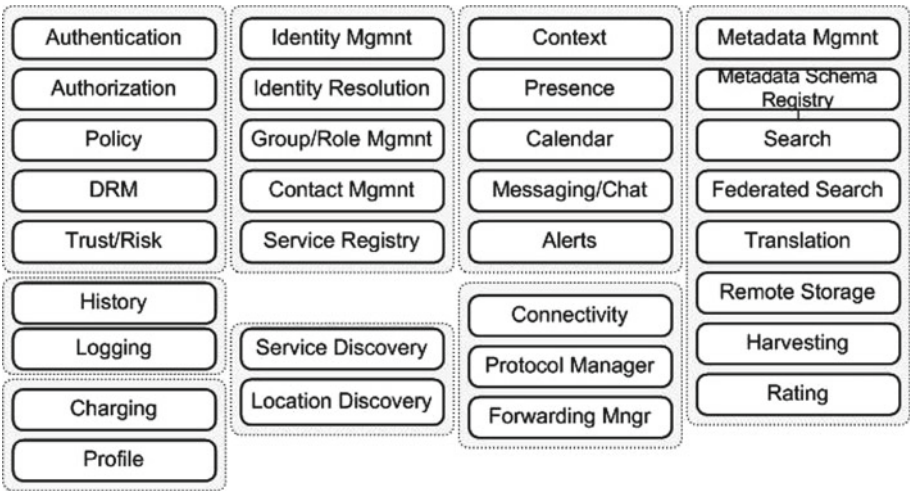

Fig. 5 ROSA framework services

an environment, the service interface may be realized using a shared implementation. For example, presence, context and messaging could all be managed using a single Jabber service. Or, authentication, authorization and accounting/charging (AAA) can be done via a single AAA service. A brief intentions of each services are mentioned in [39]. This is by no means an exhaustive list of services that can be made available, as the service oriented approach facilitates adding more or deducting unused services according to requirements. The core task of creating such a framework is to identify a broad set of services that need to be defined (called 'factoring services'). We consider factoring to be an ongoing process as experience informs the choice of services, identifies shortcomings and indicate the services that require creation, discount, splitting or joining. A service is a pattern that can used to solve a specific problem and can be defined with in a framework at different levels [41]. While this pattern can be defined at various degrees of granularity [38], we propose them as a definition of function and scope. The functional focus provides the capacity to be specific about the range of expected behaviour of individual service, while being agnostic with regards to the implementation details or design of solutions. While this definition is far from sufficient to implement a network architecture, it provides as a starting grid to more detailed specifications and a reference model. From the abstract models of services, it is possible to derive XML schemas that define data to be exchanged. This approach enables specific SOA standard based definition for services to be implemented (as a Web-Service, for example). Such a framework is realized in an application with an interface to access a service that has commonly agreed operation definitions (e.g. Web Service Definition language, WSDL) and data structures (e.g XML schemas) [42].

There are different methods to realize SOA in communication systems. Web-Services represent one important approach and is the most adopted and widespread within the IT industry. There are various other Middleware (OMG CORBA, MSDCOM etc.) that can be used for such abstraction, but Web-Services have marked advantages like being much more loose coupled, dynamic and adaptable to change. Besides, it supports and open way to develop specifications and using technology via a broad consortia, which takes into account the stakeholder interests.

The world wide consortium (W3C) defines Web-Services as:

> A Web-Service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the Web service in a manner prescribed by its description using Simple Object Access Protocol (SOAP) messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards.

Web-Services provide a uniform way of describing components or services with in a network, locating them and accessing them. Web-Service specifications define formats and protocols that allow services to interoperate across those vendor platforms that provide conforming implementations, either natively or by mapping them onto existing proprietary middleware offerings. The standards and specifications that are adopted are being developed in an open way through community organizations like W3C and OASIS. The process allows for a consensus based standardization and vetting by commercial interests before being accepted or approved as a standard. Web services rely on XML for the basic underlying data model, SOAP for the message processing and data model, and WS-Addressing for addressing services and identifying messages independent of transport.

SOAP is a simple and extensible XML based communication protocol that provides a way to communicate between applications running on different operating systems, with different technologies and programming languages. With SOAP, the underlying transport might change as the message is routed between nodes, even the mechanism selected for each hop may vary as required. An important facility is the feature that the messages can be routed based on the content of the headers and the data inside the message body. SOAP forms the messaging framework of ROSA, owing to these attributes. Another specification, WS-Addressing provides an interoperable transport independent way for identifying message senders and receivers associated with a message exchange. Besides securing end-to-end endpoint identification in messages, this specification enables messaging systems to support message transmission the networks that include middleboxes like endpoint managers, firewalls, gateways etc. Further details of these specifications are available at [33,34].

The above mentioned services can take advantage of the Service Descriptions framework available in WS which defines metadata that fully describes the characteristics of a service that are deployed in the network. This metadata provides the abstract definition of the information necessary to deploy and interact with a service. In Web-Services, Web Services Description language (WSDL) is most widely used for describing metadata for Web-Services, i.e, what a Service can do. It offers a standard language-agnostic view of services offered by a Web-Service. WSDL is an XML format for describing services as a set of endpoints that operate on messages containing either document-oriented or procedure oriented information and is well suited for the ROSA framework.

A Policy service can use the WS-Policy [43], an extensible framework for Web-Services constraints and conditions allowing for a clear and and uniform expression of of the available options. The *Service Discovery* module can query over metadata that describes services. This metadata should be searchable and discoverable for users to find and use services. Hence, aggregation and discovery services for metadata which in turn becomes a repository or registry for services are very useful, if not unavoidable. Universal Description Discovery and Integration (UDDI) is the most common used specification for a Web-Service registry.

As can be realized at this point, the Service Oriented Architectural approach for Networking through the ROSA framework can be realized using the various Web-Services specifications (WS-*). The next step is to select a subset of the services and create a basic profile, which can bootstrap into a simple functioning model using Web-Services. This is a future exercise related to this proposal.

6.4 Comparison with Other Approaches

A clear study of the drawbacks of such an approach needs to be carried out, to propose improvements to the architecture at initial stages. Coupling of numerous independent services which communicate among themselves locally or over a network will generate a large quantity of messages. The quantity of messages increases significantly with the number of services utilized indicating scalability issues. This might impose a limit on the modularity of services, as smaller services imply greater flexibility but generate higher overhead traffic. We can address this complexity partly through the use of RM which becomes the only service that each of the other services interact with, thus reducing inter-service messaging. The cost, however, is a significant complexity in the RM to correctly fit the numerous services into a logical solution for different contexts.

Efficiency for small set of services compared to legacy architectures will be less, using our approach. However, the advantages of SOA will be inherited into the proposed architecture. The integration of components within heterogenous environments or dynamically changing component configurations is best addressed using our architecture. SOA and Web-Services offer potentially significant benefits to large service sets that undergo frequent change and facilitate reusability. Currently, the XML footprint and parsing cost at both ends of a message exchange does take up time and resources. With high performance as the criterion, Web-Services might not be as efficient as current architectures. Binary XML for interchange can improve the performance, but it is yet to be standardized.

7 Conclusions and Future Work

For a 5th generation network, the principles and paradigms extend beyond bandwidth or ubiquitous computing. In such a network, every device and service is immersed in an environment of usable services, transparent to the location, be it local or remote. Network communication becomes an enabler or a pure service, where the focus shift from connectivity to usage scenarios. Processing power on demand, expandable storage, guaranteed service qualities etc will become the norm. For such a network, it is necessary that applications and users should not have to deal with low level infrastructure attributes, but an high level abstraction of such attributes as usable services. The architectural principles proposed in this article is a step towards such a networking vision.

We introduced a new architecture that treats network resources and service resources in the same way by applying service oriented principles. In addition to hiding heterogeneity and respecting business borders, this approach gives enough flexibility to migrate from today's internet towards a network of future *Internets*.

Acknowledgments The author would like to thank Prof. Frank Reichert (University of Agder, Norway) for his valuable feedback and insights related to this article.

References

1. Carr, C. S., Crocker, S. D., & Cerf, V. G. (1970). Host-host communication protocol in the arpa network. In *AFIPS '70 (Spring): Proceedings of the May 5–7, 1970, spring joint computer conference* (pp. 589–597). New York, NY: ACM.
2. Clark, D. (1988).The design philosophy of the darpa internet protocols. In *SIGCOMM '88: Symposium proceedings on Communications architectures and protocols* (pp. 106–114). New York, NY: ACM.
3. Handley, M. (2006). Why the internet only just works. *BT Technology Journal, 24*(3), 119–129.
4. Blumenthal, M. S., & Clark, D. D. (2001). Rethinking the design of the internet: the end-to-end arguments vs. the brave new world. *ACM Transactional Interet Technology, 1*(1), 70–109.
5. Carpenter, B., & Moore, K. (2001). Connection of ipv6 domains via ipv4 clouds. RFC 3056 (Proposed Standard), Internet Engineering Task Force, Feb. 2001. [Online]. Available: http://www.ietf.org/rfc/rfc3056.txt.
6. Gregg, M., & Watkins, S. (2006). *Hack the stack: Using snort and ethereal to master the 8 layers of an insecure network*. Rockland, MA: Syngress Publishing.
7. Ahlgren, B., Eggert, L., Ohlman, B., & Schieder, A. (2005). Ambient networks: Bridging heterogeneous network domains. In *Proceedings of 16th annual IEEE international symposium on personal indoor and mobile radio communications (PIMRC)* (Vol. 2, pp. 937–941).
8. Scott, J., Crowcroft, J., Hui, P., & Diot, C. (2006). Haggle: A networking architecture designed around mobile users. In *Proceedings of the third annual conference on wireless on-demand network systems and services* (pp. 86, 78).
9. Aguiar, R. L. (2008). Some comments on hourglasses. *SIGCOMM Computational Communications Review, 38*(5), 69–72.
10. Nikander, P. (2007). The host identity protocol (hip): Bringing mobility, multi-homing, and baseline security together. In *Security and privacy in communications networks and the workshops, 2007* (pp. 518–519). *SecureComm 2007. Third international conference*.
11. Clark, D., Wroclawski, J., Sollins, K., & Braden, R. (2005). Tussle in cyberspace: Defining tomorrow's internet. *Networking, IEEE/ACM Transactions, 13*(3), 462–475.
12. Mahonen, P., Riihijarvi, J., Petrova, M., & Shelby, Z. (2004). Hop-by-hop toward future mobile broadband ip. *Communications Magazine, IEEE, 42*(3), 138–146.
13. Niemegeers, I. G., & Groot. S. M. H. D. (2002). From personal area networks to personal networks: A user oriented approach. *Wireless Personal Communications, 22*(2), 175–186.
14. Lua, E. K., Crowcroft, J., Pias, M., Sharma, R., & Lim, S. (2005). A survey and comparison of peer-to-peer overlay network schemes. *IEEE Communications Surveys and Tutorials, 7*, 72–93.
15. Samad, M., Yusuf, F., Hashim, H., Mahfudz, M., & Zan, M. (2002). Deploying internet protocol version 6 (ipv6) over internet protocol version 4 (ipv4) tunnel. In *Research and Development, 2002. SCOReD 2002. Student Conference* (pp. 109–112).
16. Balakrishnan, H., Lakshminarayanan, K., Ratnasamy, S., Shenker, S., Stoica, I., & Walfish, M. (2004). A layered naming architecture for the internet. In *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications* (pp. 343–352). Portland, Oregon, USA: ACM.
17. Moskowitz, R., & Nikander, P. (2006). Host identity protocol (hip) architecture. RFC 4423 (Informational), Internet Engineering Task Force, May 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4423.txt.
18. Rescorla, E. (1999). Diffie-hellman key agreement method. RFC 2631 (Proposed Standard), Internet Engineering Task Force, Jun. [Online]. Available: http://www.ietf.org/rfc/rfc2631.txt.
19. Kent, S. (2005). Ip encapsulating security payload (esp). RFC 4303 (Proposed Standard), Internet Engineering Task Force, Dec. [Online]. Available: http://www.ietf.org/rfc/rfc4303.txt.
20. Koponen, T., & Gurtov, A. (2005). Application mobility with host identity protocol. In *Proceedings of NDSS wireless and security workshop*.
21. Novaczki, S., Bokor, L., & Imre, S. (2007). A hip based network mobility protocol. In *SAINT-W '07: Proceedings of the 2007 international symposium on applications and the internet workshops* (p. 48). Washington, DC: IEEE Computer Society.
22. Hui, P., Chaintreau, A., Gass, R., Scott, J., Crowcroft, J., & Diot, C. (2005). Pocket switched networking: Challenges, feasibility, and implementation issues. In *Proceedings of the workshop on autonomic communications*.
23. Su, J., Scott, J., Hui, P., Crowcroft, J., de Lara, E., Diot, C., Goel, A., Lim, M., & Upton, E. (2007). *Haggle: Seamless Networking for Mobile Applications.*, Ser. Lecture Notes in Computer Science (Vol. 4717, pp. 391–408). Berlin: Springer. [Online]. Available: http://www.cl.cam.ac.uk/~ph315/publications/haggle-ubicomp2007.pdf.

24. Niebert, N., Schieder, A., Zander, J., & Hancock, R. (2007). *Ambient networks: Co-operative mobile networking for the wireless world*. London: Wiley Publishing.
25. Ahlgren, B., Eggert, L., Ohlman, B., Rajahalme, J., & Schieder, A. (2005). Names, addresses and identities in ambient networks. In *DIN '05: Proceedings of the 1st ACM workshop on Dynamic interconnection of networks* (pp. 33–37). New York, NY: ACM.
26. Koponen, T., Chawla, M., Chun, B. -G., Ermolinskiy, A., Kim, K. H., Shenker, S., & Stoica, I. (2007). A data-oriented (and beyond) network architecture. *SIGCOMM Computational Communications Review, 37*(4), 181–192.
27. Bannazadeh, H., & Leon-Garcia, A. (2007). On the emergence of an application-oriented network architecture. In *Proceedings of the IEEE international conference on service-oriented computing and applications* (pp. 47–54). IEEE Computer Society.
28. Knightson, K., Morita, N., & Towle, T. (2005). Ngn architecture: Generic principles, functional architecture, and implementation. *Communications Magazine, IEEE, 43*(10), 49–56.
29. Atkinson, R., Bhatti, S., & Hailes, S. (2009). Ilnp: Mobility, multi-homing, localised addressing and security through naming. *Telecommunication Systems, 42*(3), 273–291, Dec. [Online]. Available: http://dx.doi.org/10.1007/s11235-009-9186-5.
30. O'Dell, M. (1996). 8+8—An alternate addressing architecture for ipv6, 10.
31. Nordmark, E., & Bagnulo, M. (2009). Shim6: Level 3 multihoming shim protocol for ipv6. June. [Online]. Available: http://tools.ietf.org/html/rfc5533.
32. Trossen, D., Briscoe, B., Sollins, P. M. K., Zhang, L., Mendes, P., Hailes, S., Jerman-Blaciz, B., & Papadimitrou, D. (2009). Eiffel report: Starting the discussion. EIFFEL think tank, Technical Report.
33. OASIS. Oasis soa commitee. [Online]. Available: http://www.oasis-open.org/committees/tc_cat.php?cat=soa.
34. W3C. W3c web services architecture. [Online]. Available: http://www.w3.org/TR/ws-arch/.
35. Linthicum, D. S. (2000). Enterprise application integration. Essex, UK: Addison-Wesley Longman Ltd
36. Corporation, C. S. (2000) 13th Ann. critical issues of is management survey. Computer Sciences Corporation, Technical Report [Online]. Available: www.csc.com/survey.
37. Peltz, C. (2003). Web services orchestration and choreography. *Computer, 36*(10), 46–52.
38. Weerawarana, S., Curbera, F., Leymann, F., Storey, T., & Ferguson, D. (2005). *Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging, and More*. Prentice Hall PTR, march [Online]. Available: http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20&path=ASIN/0131488740.
39. Kumar, R., Haber, A., Reichert, F., & Yazidi, A. (2010). Towards a relation oriented service architecture. In *Proceedings of 2010 second international conference on communication systems and networks (COMSNETS 2010)*, no. CFP1031F-CDR. Institute of Electrical and Electronics Engineers (IEEE), 2010, iSBN: 9781424454891.
40. Bellavista, P., & Corradi, A. (2006). *The handbook of mobile middleware*. Boca Raton: Auerbach Publications.
41. Nicholls, P. (2009). Enterprise architectures and the international e-framework. e-Framework Partnership for Education and Research, Technical Report 1.3 Final, Jul. [Online]. Available: http://www.e-framework.org/Portals/9/docs/EAPaper_2009-07.pdf.
42. Christensen, E., Curbera, F., Meredith, G., & Weerawarana, S. (2001). Web service definition language (wsdl). World Wide Web consortium, Technical Report, NOTE-wsdl-20010315, March. [Online]. Available: http://www.w3.org/TR/wsdl.
43. Vedamuthu, A. S., Orchard, D., Hirsch, F., Hondo, M., Yendluri, P., Boubez, T., & Yalçinalp, mit (2007). Web services policy framework (wspolicy). September. [Online]. Available: http://www.w3.org/TR/ws-policy.

Author Biography

Ram Kumar received his Masters degree from University of Liveprool, UK. Prior to that, he has worked with companies like D-Link and academic startups like PicoPeta Simputers before commencing his research at Agder Mobility Lab, University of Agder, Norway in 2006. His areas of interest include Next Generation Networks, Network architectures, Mobile and Pervasive computing, and Service Oriented Architectures.