ANA Project

Autonomic Network Architecture

Sixth Framework Programme
Priority FP6-2004-IST-4
Situated and Autonomic Communications (SAC)

Project Number: FP6-IST-27489

Deliverable D2.1
First draft of routing design and service discovery

Version 1.0

ANA Project

Autonomic Network Architecture

Project Number	FP6-IST-27489
Project Name	ANA - Autonomic Network Architecture
Document Number	FP6-IST-27489/WP0/D2.1
Document Title	First draft of routing design and service discovery
Workpackage	WP2
Editors	Serge Fdida (UPMC), Ioannis Stavrakakis (NKUA)
Authors	Konstantinos Oikonomou (NKUA) Sylvain Martin, Guy Leduc (ULg) Martin May, Simon Heimlicher (ETH) Yosra Barouni, Prométhée Spathis (UPMC) Stephan Schmid, Rolf Winter(NEC), Christophe Jelger(UBasel), Manolis Sifalakis(ULanc), James P.G. Sterbenz (ULanc and KU)
Reviewers	Christophe Jelger(UBasel), Thomas Plagemann, Karl-André Skevik (UiO)
Dissemination level	Public
Contractual delivery date	31st December 2006
Delivery Date	December 2006
Version	1.0

Abstract:

This document presents the First draft of routing design and service discovery for the ANA architecture

Keywords:

Autonomic, Routing, Service Discovery

Executive Summary

The objective of this deliverable is to present the critical routing and service discovery functionalities within the ANA architecture. Routing provides connectivity and Service Discovery gives access to content and services, therefore, they are instrumental in the ANA architecture, in a dynamic and distributed environment. The main challenges are highlighted and basic solutions are presented. This analysis is leading to emphasis the solutions that will contribute to the ANA architecture.

The first part (section 2) of the document briefly recaps the rich diversity of routing protocols and mechanisms that exist today: the goal is to stress the fact that a new network architecture should not impose any “one-size-fits-all” communication scheme (i.e., for routing, addressing, naming, etc). A flexible routing framework based on elementary components that can be re-combined on-demand to create ad-hoc instances of routing protocols is then presented. The long-term motivation is to understand how existing routing protocols can be automatically instantiated with the help of a generic set of elementary building blocks.

The second part (section 3) of the document provides an introduction of service discovery as foreseen in the ANA project. Besides defining potential uses of service discovery within ANA, the document briefly summarizes existing architectures and schemes for service discovery in order to highlight requirements and open challenges. A service discovery framework for ANA is then presented, and early results based on simulations are discussed. This section finally presents a lightweight platform for service discovery that could potentially be used inside ANA.

The final section of the document discusses some possible interactions between routing and service discovery and proposes some areas where the operation of these two components could be merged or optimized via closely coupled operation.

Table of Contents

1 INTRODUCTORY

Work Package 2 focuses on the basic elements and mechanisms needed to establish paths between two or more communicating entities. This includes naming, addressing, forwarding, and routing schemes for point-to-point, group and overlay-like communications. In the context of autonomic communications, it is also important for individual network nodes to be able to automatically bootstrap (*auto-configure* and *self-associate*) in a new network context, or for the overall network to autonomously organise and configure itself (*self-organise*). Therefore this WP deals with the issues of discovering service and network functionality, ad hoc integration of individual nodes in the network as well as dynamic formation of complete networks (and this includes operations such as cloning and partitioning as well as merging of, potentially mobile, networks). At the same time a parallel goal of this WP is to investigate ways of facilitating such enriched and dynamic operations, within the network subsystem of the autonomic network node.

In this deliverable, we present the first elements of the routing and service discovery design. We address the development of a routing framework, as well as the design of service discovery and service advertisement mechanisms that will capitalize on the innovative schemes of ANA (mainly routing and self- abilities of the network). These mechanisms will allow the dynamic sensing of the network environment capabilities as well as the discovery of network functionalities.

1.1 Scope of Deliverable

The objective of this deliverable is to discuss the critical routing and service discovery functionalities within the ANA architecture. It is leading to the study of new solutions that will contribute to the ANA architecture.

1.2 Structure of the document

In section 2.2, some existing routing protocols are described to highlight the variety of existing solutions. This part is not supposed to be exhaustive but should represent a fair sample. Section 2.3 provides a first analysis of the routing family with respect to the ANA requirements. Section 2.4 introduces the main routing framework for the ANA architecture.

An attempt for definition of (network) services and service discovery is presented in Section 3.1. In 3.2 existing service discovery architectures are presented along with a proxy services in transit networks. Section 3.3 presents the potential use of service discovery in other tasks in the ANA project and finally a possible framework is presented in Section 3.4.

2 ROUTING

2.1 Introduction

The last decade has produced a tremendous accumulation of knowledge in the area of networking, supporting the enormous success of the Internet that became a commodity network. As such, innovation inside the network itself is rendered difficult, as the Internet technology is the foundation of many business and consumer applications. This is certainly the reason behind today's important activity in overlay networks as innovation is still possible at the network edges.

However, connectivity is the main service provided by a network, and supported by the routing protocols. They were first designed to address the problem of forwarding a packet towards its destination with a set of functionalities. Since a few years, networking has enlarged its scope with the emergence of a variety of new technologies like ad-hoc networks, sensor networks, DTNs(delay-tolerant networks) and overlays. These technologies, in addition to new routing requirements (multicast, QoS, mobility, large-scale) have pushed the routing principles closer to their limits. Moreover, in order to support new functionalities, small improvements have been added to the architecture at a cost of an increased complexity and less efficiency. We claim that it is time to re-assess the relationship between the various components of a routing protocol in order to appropriately address these above-mentioned challenges, recognizing that today's improvements incorporated in the architecture will suffer to survive the network evolution.

Similarly, we have to study the development of efficient algorithms for service discovery (pull mode) and service advertisement (push mode) as well as to explore the design space between these two modes that could lead to results of great importance for autonomic overlay and peer-to-peer networks. We first focus on the algorithmic aspects of the aforementioned mechanisms and the design of a lightweight active platform for services and communities discovery. More specifically, we consider - the design of a joint routing and naming mechanism to facilitate service and network function discovery, - the interactions between addressing, naming, routing and the service discovery process will be studied and spontaneous discovery mechanisms will be developed, based on Distributed Hash Table (DHT) abstractions and regular addressing structures, - the efficiency of service advertisement with regard to scalability and the induced overhead. Since the information contained in control data is needed in different parts of the network at different levels of granularity, mechanisms such as filtering, diffusion, aggregation and other limited information dissemination strategies could be employed to reduce the associated overhead.

Typically, a routing protocol consists generally in three elements: a set of routing protocols that allow actors to collect and distribute routing information, a routing information base that can be centralized or distributed, partial or complete, and a routing algorithm that uses this base and the information stored in it to derive or compute routes. The control-plane routing activity is then typically used to setup states inside the data-plane where packet forwarding takes place. However with emerging routing paradigms such as for e.g., wireless ad hoc, sensor, and delay-tolerant networks, the routing and forwarding mechanisms are not always clearly separated as they are in the traditional IP world. For example in reactive routing protocols for wireless ad hoc networks, the operations of routing and forwarding are closely coupled: route discovery is triggered by the absence of an adequate forwarding entry, and during the route discovery procedure packets are temporarily buffered and not dropped (although there is no route yet and no guarantee that a route will be discovered). In other cases such as in sensor networks, forwarding may depend on the data contained in the packet being processed, for example if data aggregation is performed before (re-)transmission. Also note that in addition to purely algorithmic concerns, routing also encompasses closely related aspects such as addressing (and to some extent, naming) since routing performance usually depends on address allocation and the ability for protocols to efficiently aggregate routing information.

Hence in order to take into account the multi-facet nature of routing (in a broad sense), in ANA we explore a flexible routing framework that does not only focus on algorithmic aspects such as information dissemination and discovery, but also explicitly includes closely related concepts such as forwarding, addressing, and naming. The main idea is to identify a set of generic and atomic components that can be re-arranged and parameterized on-demand in order to instantiate different types of routing protocols. The main motivation is to avoid that each routing protocol re-implements everything from scratch, with basic features such as e.g., neighbor discovery or message framing and parsing being provided by a generic library of routing functions. Existing components can then be arranged in arbitrary ways to make up new protocols tailored towards a specific target scenario, the challenge in ANA being that this procedure is performed in an autonomic way. In addition to providing a component-based routing framework, a key feature of ANA is also to provide interoperability between the various routing entities that will have, at some point in time, to cooperate. Again the challenge in ANA is to achieve interoperability in an autonomic manner with e.g., generic translation and encapsulation mechanisms.

2.2 Analysis of existing Routing Approaches

A wide variety of routing approaches exist, each one designed to handle the special network characteristics and operational needs it was designed for, in a near-optimal fashion. Although the actual implementations of these routing protocols differ significantly in many aspects, a few common conceptual components exist over the whole range of protocols, whereas not all of these components need to be actually present simultaneously. A routing protocol’s essential task is to build a forwarding table so that the forwarding process can relay an incoming packet to the appropriate outgoing interface and do some packet processing if necessary. The table lookup process is certainly one of the most consuming tasks to handle (as on a per packet basis) and the size of the table matters and directly impacts scalability. However, this table does not have to reflect the full network topology not to be maintained proactively as will be discussed in the following subsections. A hierarchical address allocation approach can also be used in order to reduce the size of the table by aggregating forwarding entries.

In the next subsections, we present the basic operation of some routing protocols in various networking environments. Our goal is not to provide an exhaustive survey of routing protocols but to expose the variety of situations that a component-based routing framework will have to address.

2.2.1 Routing in the Internet

2.2.1.1 Basic operation

In today's Internet, routing is achieved via a complex multi-tier hierarchy of routing domains. That is, there are multiple levels at which routing is performed and in practice each routing level has quite different operational requirements and performance objectives. In particular, the practices and rules for deploying and running routing protocols and routing policies can radically differ between these routing levels, along with performance metrics, reliability requirements, and management platforms. Networks also highly differ in terms of number of routers and switches, physical connectivity, deployment scope, link layer and hardware technology (for both routing devices and communication links), and scalability issues (routing load, table sizes, routing updates). Actually the only (purposely designed) common denominator between these routing levels is the IP global addressing space which federates routing at the network layer (and this is what IP was designed for). Note that the goal of this section is not to review routing protocols, but rather to highlight the challenges faced by Internet routing and the extreme variety of networks that compose the Internet.

At the finest level with home or small networks hosting a few devices, routing (if any) is easy to manage, suffers no scalability constraints, and may be statically and manually configured by a (skilled) human user. Connectivity is typically very low, with usually a single link and default route towards a network provider. Processing requirements (CPU power, bandwidth, forwarding speed) are usually low. However, the average user has very low or no technical skills to configure, daily manage, or repair routing so from an autonomic perspective routing has to self-configure and self-repair. To cite just a few, protocols such as DHCP, NAT, and IPv6 prefix delegation partially fulfil these technical goals.

At the next routing level with university and medium to large company networks, routing has to cope with a few dozen routers and up to a few thousand hosts, possibly private networks (with NAT), with typically multiple IP subnets and virtual private networks (VPNs). For reliability purposes, such networks have redundant links to cope with link or router failures. To achieve a better Internet connectivity via multi-homing, there may be multiple up-links towards multiple network providers. Dynamic routing is achieved autonomously via interior gateway protocols (IGP) such as RIP [58,59], OSPF [56,57], or IS-IS [62]. Performance requirements are not very high with links of up to 1 or 10 Gb/s and modest routing table sizes (e.g. up to a few hundred entries). Some of these networks might participate in BGP routing with usually a very low number of peerings (i.e. typically one per network provider).

At the coarsest level, very large networks grouped in autonomous systems (ASs) are interconnected via BGP [61]. The largest ASs host tens of thousand routers running the OSPF or IS-IS routing protocols with very high performance requirements and large volume of traffic. Connectivity within an AS might be high (e.g. highly meshed structure) with multiple redundant paths and thus large routing table sizes. Connectivity at the BGP level ranges from very few connected peers (i.e. degree smaller than 5) to a very little number of highly connected (Tier-1) peers with more than one thousand adjacencies. The largest routers of the default-free zone have up to 200,000 FIB entries. BGP routing is steered by subtle business relationships where competitors have to collaborate in order to guarantee global reachability among all Internet hosts. That is, technical considerations are not necessarily the main driver of the daily operational decision making process.

2.2.1.2 Limitations, problems, trends

As briefly described in the previous paragraph, routing in the Internet ranges from networks with a small number of low-connected devices to very large networks with high performance requirements and complex business relationships. Surprisingly, and beside the fact that there no “one-size-fits-all” routing protocol, the vast and diverse networking conditions of today’s Internet are covered by a very low number of routing protocols, namely RIP, OSPF, IS-IS and BGP. However, this high flexibility is achieved at the very high cost of meticulous and empirical human configuration and tuning of routing protocols on a case-by-case manner. This intense and highly skilled human intervention has to deal with a large number of protocol parameters, routing systems and interfaces, networking conditions, and with the lack of built-in and standard mechanisms to assess the performance of the network during the configuration and tuning phases.

In addition, the daily management of large networks is highly time consuming and requires an explicit coordination between the operators of independently managed networks. For example, it is almost impossible for a network operator to identify the exact cause of a network outage if the failure occurs in another operator’s network. Apart from bare tools like traceroute and ping, the Internet has no large-scale built-in protocol to help debugging the network. This frustration also reaches end users who have no easy way to find out why a service or a host is not reachable.

From an architectural perspective, the management complexity of the Internet is increased by the fact that a unique global addressing space is shared by all Internet users. This requires a meticulous coordination to allocate addresses (i.e. between IANA, RIRs, and LIRs) and a global, implicit, and critical trust relationship which dictates that “no one should use addresses allocated to someone else”. In addition, global addressing implies global visibility (although not necessarily reachability, due to firewalls) which greatly increases the risks related to attacks and worms. The main and growing consequence of the Internet "global and default-on" addressing model is the wide spread use of NAT, which makes address allocation a local business and makes reachability an explicit feature.

2.2.1.3 New requirements

Over the past ten years, growing networking requirements such as mobility and multi-homing are putting some extra burden on Internet routing. Mobility at the network layer increases path latency because of triangular routing, and it breaks Internet transparency and end-to-end connectivity by introducing redirection points (i.e. home agents). Multi-homing forces operators to de-aggregate network prefixes and inject extra information into the routing system, thus increasing the size of routing tables with a high number of long prefixes.

Mobility at the network layer has also exacerbated the fact that IP addresses are (misused) by transport and application layers to identify connections. This layer violation increases the complexity and deployment of mobility protocols (e.g. [63]), which have to present a long-lived identifier to the higher layers. Multi-homing suffers from the same problem and solutions being currently developed (e.g. [64]) also have to provide a unique identifier to the transport and application layers. The identifier/locator split is also addressed in a long-term manner at the IETF (i.e. [65]).

2.2.1.4 Requirements for ANA routing

From the current situation of Internet routing as briefly described in the previous paragraphs, we can derive some requirements for the routing of our future autonomic network architecture. Note that traditional requirements such as resilience, convergence, security, and trust are also considered.

i) No one-size-fits-all + adaptability.

There should be no "one-size-fits-all" routing protocol, especially since emerging networking paradigms (ad hoc, sensors, DTN) have very different routing requirements than the current Internet (as described in other sections in this document). In particular, some routing protocols are optimized to handle mobility or sporadic connectivity while others are designed for very large networks with severe scalability constraints. In addition to supporting multiple routing schemes, networking nodes should be able to automatically activate appropriate routing protocols in an ad hoc manner, i.e. according to networking conditions and performance requirements. A more aggressive (and resource consuming) strategy is to run multiple routing schemes in parallel and dynamically switch packets towards the most dependable routes or according to some performance criteria.

ii) Self-configuration and self-optimization.

To reduce the burden on network management, routing protocols should be able to self-configure and constantly self-optimize according to high-level directives and policies. Dedicated instances should constantly monitor the performance of routing protocols and possibly tune their parameters according to well-defined rules and previous experience gained via learning algorithms. We here note that this higher flexibility and adaptability might initially come at the cost of increased complexity and reduced performance (e.g. in terms of packet forwarding speed).

iii) No global and unique addressing scheme.

The current experience with global IP addressing and the trend towards NAT shows that independently and locally managed addressing spaces is an appealing networking model. The federation of heterogeneous networks hence become critical and, in contrast to what was done with IP, will be achieved without introducing a unique and global addressing space. The alternative is to introduce multiple and potentially different layers of addressing and naming with subsequent resolution steps along a routing and forwarding path. This approach generalizes the basic Internet concept of network federation (i.e. where IP federates multiple link layers) to a model with multiple federation levels of heterogeneous addressing and naming schemes. In certain cases, this strategy will also require the introduction of generic translation mechanisms to interconnect networks with incompatible addressing schemes.

2.2.2 Routing in (Mobile) Ad hoc Networks

2.2.2.1 Basic Assumptions and Operation

The concept of ad hoc networking can be found in a vast amount of possible scenarios. Those range from disaster response, vehicle-to-vehicle communication, community networks, smart spaces and e-learning to battlefield networking. All these scenarios have different characteristics and requirements which potentially vary enormously from application to application. Independent from applications and scenarios there are some key characteristics which can be found in all ad hoc networks although with varying impact on the networking performance. The key property is that ad hoc networks are self-organizing, infrastructure-less networks. This lack of infrastructure adds a high degree of complexity as centralized approaches might not be feasible to be realized. Many problems that can easily be solved using some central entity such as a server or a high performance communication backbone need to be solved differently from existing solutions that assume an infrastructure. Such problems which include for example unique (IP) address assignment, object and service location or synchronization have to be solved in a distributed fashion in ad hoc networks. In addition, nodes in ad hoc networks are typically mobile resulting in topological changes and disconnection. The shared wireless medium is error-prone with fast, time-varying channel characteristics and only provides a relatively low bandwidth. The combination of these problems makes routing protocol design challenging. A routing architecture must therefore provide a flexible framework giving sufficient support to overcome the challenges just described. The devices used in such environments are also likely to be battery-driven and therefore energy management can become an important issue. Clearly, ad hoc networks can be highly dynamic and resource constrained, resulting in a multitude of problems which can arise in such environments. Fundamental principles applied today in most routing protocols need to be revised such as strict layering of protocols or the end-to-end principle.

2.2.2.2 Limitations and requirements

A plethora of ad hoc network routing protocols have been designed. The focus of this section is not to provide a complete list of state-of-the-art routing protocols that have been designed but to identify the problems that they try to address. From these problem formulations a set of requirements are derived that any new routing architecture needs to meet. A popular classification of ad hoc routing protocols is to distinguish between proactive/table building protocols such as DSDV [66] and reactive/on demand protocols such as DSR [67]. Other distinguishing factors are whether the routing protocol establishes a structure/hierarchy or whether it routes amongst equal peers (flat). Many other classification criteria exist but the sheer amount of different approaches and classifications demonstrates that routing in such dynamic networks is a very challenging task and that ad hoc networks can be very heterogeneous depending on factors such as end device capabilities, radio technology, node mobility or communication patterns. A one-fits-all solution to the routing problem can therefore not be found. It seems a single routing protocol can only satisfy a set of ad hoc networks having characteristics within certain bounds. In the ad hoc networking context some requirements seem to be especially important:

i) Adaptability/Re-configurability

Changes of information relevant for routing can be frequent in ad hoc networks. Therefore a routing protocol needs to react timely to such changes. But current, statically parameterized protocols fail adapt efficiently. As an example, AODV [54] sends out a Hello message once every second. In static scenarios this might be far too much overhead. Therefore it should be possible to re-configure protocols on-demand to make them work efficiently in a given environment.

ii) Security/Trust

Ad hoc network are a cooperative environment. Every node in the network can potentially become a router for every other node. Such an environment leaves some space for malicious users trying to disrupt the routing system. It therefore is important to deal with this issue in a cooperative, autonomic way. This issue not only concerns routing, and with ANA it is dealt with up at the compartment level [91].

iii) Modularity

Re-configuration as described above might not be enough. A routing framework should be modular enough to change the routing strategy when needed. As an example, if the mobility of the network becomes too high a sophisticated routing protocol might not perform very well. The only way to deliver packets successfully might be to start broadcasting the packets. Such a change in strategy should be supported by a routing architecture.

iv) Robustness/Resilience

In ad hoc networks node failure might be a very common phenomenon. That might simply be because of battery depletion or a user switches his/her device off. Mechanisms for resilience and robustness against failure need to be in place to compensate for such outages. An architectural solution or architectural support for this problem is preferable as otherwise every single protocol needs to build robustness mechanisms from ground up.

The ANA routing framework needs to be designed with such requirements in mind. Ad hoc networks are very demanding environments, especially for routing protocols. Having strong architectural support would make routing protocol design easier and the protocols themselves could be more efficient.

2.2.3 Routing in Sensor Networks

2.2.3.1 Basic Assumptions and Operation

The field of micro-electro-mechanical-systems (MEMS) has shown great advances recently and, combined together with wireless communications and digital electronics, has enabled the development of low-cost, low-power, multifunctional sensor nodes that are small in size and communicate over short distances. The possibility of these sensor nodes to be networked together over a wireless medium, and to provide, through collaborative effort, an overall result of their sensing functionality is raising a whole, new field of research for networking engineers and researchers (Wireless Sensor Networks), [68]. Among the challenges that WSNs have to overcome are the constraints on the energy resources and the bandwidth of the sensors, as well as on node mobility and flexibility regarding routing.

Energy depletion of the tiny sensor nodes is the main resource bottleneck for the operation of routing protocols for WSNs, [69]. Furthermore, other design challenges to meet the extreme hardware limitations of sensor nodes include reducing each sensor’s active duty cycle and minimizing data communication over wireless channel and thus maximizing total network lifetime. Further limitations of routing for WSNs are the need for robustness against dynamic environmental changes and for network scalability to thousands of nodes.

Concerning routing in WSNs, the raised challenges are rather different than the ones found in the traditional ad-hoc networks and wireless networks. For instance, no global addressing scheme is easy to be supported; the data that comes from different regions are destined to specific sinks; all data need to be processed by each node so that redundancy is reduced and energy is saved; sensor nodes are generally stationary so that mobility cannot be used as a means to carry data.

The routing protocols in WSNs could be categorized into the data-centric, hierarchical and location-based ones based on their functionality, [76]. Data-centric protocols are query-based and depend on the naming of the demanded information, so that redundancy is limited. The most straightforward idea for routing data might be adopted by the traditional flooding and gossiping techniques, [74]. According to these techniques, every node receiving data from another broadcasts it to all (or randomly selected set of) its neighbors and this process continues until data is delivered to the destination. Another family of adaptive protocols proposed, called SPIN (Sensor Protocols for Information via Negotiation), efficiently disseminates information among sensors in an energy-constrained wireless sensor network [78]. Hierarchical protocols aim at clustering the nodes so that clusterheads can aggregate data so that energy is reduced. By analyzing the advantages and disadvantages of conventional routing protocols using the model of sensor networks, some researchers have developed LEACH (Low-Energy Adaptive Clustering Hierarchy), a clustering-based protocol that minimizes energy dissipation in sensor networks in [75]. Location-based protocols take advantage of position information to forward data only in specific regions instead the whole network. MECN, [77], sets up and maintains a minimum energy network by utilizing low-power GPS; therefore, a minimum power topology for stationary nodes is found.

The aforementioned routing protocols are only indicative since there are numerous protocols proposed for sensor network. The list is not complete since the role of this section of the deliverable is not literature review (which is rather extended) but the identification of possible limitations and the derivation of the corresponding requirements for the ANA environment.

2.2.3.2 Limitations and Requirements

To cater to the peculiarities of WSNs, several routing protocols have been proposed in the literature. In accordance to the limitations discussed in the previous section, the metrics described below have been proposed as the goals that routing protocols in WSNs should satisfy:

i) Energy efficiency/system lifetime

As sensor nodes are battery-operated, routing protocols must be energy-efficient to maximize system lifetime [70]. System lifetime can be measured by generic parameters such as the time until half of the nodes die or by application-directed metrics, such as when the network stops providing the application with the desired information about the phenomena.

ii) Latency

The observer is interested in knowing about the phenomena within a given delay. The precise semantics of latency are application-dependent.

iii) Accuracy

Obtaining accurate information is the primary objective of the observer, where accuracy is determined by the given application. There is a trade-off between accuracy, latency and energy efficiency [71]. The given infrastructure should e.g. be adaptive so that the application obtains the desired accuracy and delay with minimal energy expenditure. For example, the application can either request more frequent data dissemination from the same sensor nodes or it can direct data dissemination from more sensor nodes with the same frequency.

iv) Resilience

Sensors may fail due to surrounding physical conditions or when their energy runs out. It may be difficult to replace existing sensors; the network must be resilient such that non-catastrophic failures are hidden from the application. Scalability for sensor networks routing protocols is also a critical factor. For large-scale networks, it is likely that localizing interactions through hierarchy and aggregation will be critical for ensuring scalability.

Some further routing protocol design characteristics are the need for application specific design of the protocol, [72]. Traditional networks are designed to accommodate a wide variety of applications. On the contrary, WSN networks can be tailored to the sensing task at hand. This means that intermediate nodes can perform application specific data aggregation and caching or informed forwarding of requests for data. This is in contrast to routers that facilitate node-to-node packet switching in traditional networks. Traditional networks provide large bandwidths, wall power and powerful compute elements. Sensor nodes will often be limited in one or all of these dimensions.

In conclusion, the ANA routing framework has to take into consideration the aforementioned list of requirements which may overlap with other "traditional" environments in some points (e.g., ad hoc networks). The approaches mentioned before may be helpful for efficient use of network resource (e.g., energy in ANA) and reduce the overhead of the network.

2.2.4 Routing in Delay Tolerant Networks (DTN)

2.2.4.1 Basic Assumptions and Operation

Delay-tolerant networks (DTN) constitute another class of networks where routing is a major issue. DTNs can be wireless ad hoc networks, large-scale sensor networks, or even deep-space networks based on satellites or communication devices installed on planets.

The term DTN stems from work on the Interplanetary Internet, but nowadays represents a more general type of networks with characteristics differing from those of the Internet.

The most curtailing characteristic is the disruptiveness of links and thus intermittency of connectivity. Thus, routing information has a short lifetime, possibly shorter than the lifetime of the messages to be transferred.

The use of late binding for names in DTN is shared with, although not directly based upon, the work on Intentional Naming. Here, names represent a form of query and are used specifically for anycast in order to locate nearby network services. Routing based on names is shared, to some degree, with Internet Content Routing. This work focuses on using routing on names to provide a content distribution facility for the Internet, addressing its scalability and performance.

The architectural thinking regarding interoperability and layering is guided by principles of the ARPANET/Internet. DTN gateways operate in many ways similar to Internet routers, but are adapted for use in high-delay and disconnected environments by storing messages for potentially long periods of time. Thus, routing protocols for DTNs strive to minimize delays and storage overhead while maximizing throughput.

Some protocols were proposed based on the time cost of transmission like MED, ED, EDLQ, EDAQ and LP. The solutions in this domain propose generally a cost-based approach that tries to assign costs to links and then computes min-cost paths. The main problem in these solutions is that the routing is based on a global knowledge of the network topology, which is not feasible in the ANA architecture.

A more practical approach is epidemic routing, where data is transmitted to every neighbor in range until the destination is finally reached. Even though the costs of this approach are prohibitive, under certain assumptions it is the only possible way of transmission.

A difference to ad hoc networks is that in DTNs, it is not necessarily assumed that nodes are wireless, mobile, and unreliable. For instance, the link among a lunar station and a satellite may frequently be disconnected, but these interruptions are predictable and the communication devices are extremely reliable, and thus using a concatenation of multiple TCP connections can be all that is needed.

The general approach to routing and forwarding in DTNs is to split messages into smaller chunks and transfer them independently. In the DTN research group of the IETF, the paradigm of custodial transfer is applied, which means that the responsibility to deliver a chunk of data is transferred along with the data. Hence, intermediate nodes are responsible for the delivery to the same degree as the source.

The addressing space is approximated as flat, even though the first DTNs were all implemented as overlays over existing IP networks.

There exists a reference implementation for DTN by the DTN research group. Other related papers are [79, 80,92].

2.2.4.2 Limitations and Requirements

The limitations of DTN routing are that it is typically not meant to scale to the number of nodes. Furthermore, routing in DTNs nowadays assumes delay-aware applications and ignores established requirements such as end-to-end reliability etc.

The requirements on delay for DTN routing are very low; DTN routing aims to work with the lowest common denominator of the nodes. However, most proposals assume highly cooperative intermediate nodes, which is key for custodial transfer of messages. Furthermore, vast amounts of buffers space may be required at any intermediate node depending on the cumulated bandwidth delay product of the connections crossing the node.

2.2.5 Routing in Peer-to-peer and Overlay Networks

2.2.5.1 Basic assumption and operation

Peer-to-peer overlays are networks mostly operating at the application-layer. In this kind of networks, we usually find properties required for ANA routing framework like self-organization and distribution. Peer-to-peer networks are interesting in the context of the ANA project as they assume already many functions of self-organization, dynamicity and decentralization. Therefore, it is also of interest for service discovery as it is presented later in Section 3.2.1. As the overlay networks build virtual connections on top of the physical links of the network, their topology may change all the time. Once a route is established, there is no guarantee of the validity for this route after a lapse of time.

The routing protocols used in the overlay networks can be categorized into structured and unstructured protocols. This categorisation is based on the used localisation/routing mechanism and the presence of virtual connections between peers.

The unstructured networks like GNUTELLA[51] are based on unmanaged topology. The absence of nodes structure makes the routing more complex in term of delays and overhead. Unstructured overlays use flooding, gossiping or random walks on the network graph to diffuse and retrieve information needed by nodes. The localization of content and the localization of nodes are stored in server/client systems that are unacceptable for the ANA architecture.

To overcome the limitation of the first peer-to-peer systems, DHT tables were proposed in order to structure the organization of nodes and content inside these systems. Thanks to these structuring, DHTs offer a good combination between scalability and an efficient routing scheme. A wide variety of structured routing algorithms have been proposed like Chord[83], CAN[89], Pastry[52] and Tapestry[85].

The routing in these protocols depends mainly on the geometry of routes stored by the peers. For instance, CAN organizes routes along a cube, Chord around a ring, Viceroy uses a butterfly network and Pastry uses a hybrid routing geometry combining a tree and a ring. Thanks to this geometry, DHTs allow flexibility in neighbour and route selection. Despite the progress done in the field of peer-to-peer networking, it still suffers from some problems. First, the overhead of routing and maintaining links between data and nodes increases as soon as the number of participants increases. And if we add the mobility of the users the overhead will increase hugely. The second problem is that the structured overlays were designed to operate on top of the Internet, which is a fixed architecture. Therefore, we have to think about a solution to support the mobility of users and the adaptability of the network to the persistent change of the data and nodes location.

Clearly, principles applied in the overlay networks cannot be totally adapted to the autonomic networks where topology can be very dynamic and physical constraints are very limited.

2.2.5.2 Limitations and requirements

We can summarize the main issues in the peer-to-peer networks like following:

i) Scalability and adaptability to topology changes

The overlay networks offer a good adaptability to the persistent change of a flat network topology. As the ANA routing framework needs to offer a good reactivity to the topology changes inside and outside a compartment, we have to design a routing protocol that allows local and global changes of the topology. We mean by local changes those that occur inside a compartment and global ones those that occur in the topology and the disposition of the compartments themselves. A possibility is to make the routing framework use many routing protocols in order to adapt itself to each type of change.

ii) Hosting Resources

The peers share their systems resources to route a message, send an emails or even print a file. But in an autonomic network and in ANA specifically, this means that the limited energy and probably the computation ability can be rapidly consumed. In a classical p2p network, a solution proposed to solve this problem is the deployment of distributed and dedicated servers. But in the case of the autonomic networks and specifically in the case of the ANA network, a solution based on central servers may not always be acceptable. Therefore, the routing framework should offer a mechanism to load balancing the treatments between the whole nodes of the network and consider not only the link state but also the node state before routing messages.

iii) Maintenance

We believe that the ANA routing framework needs to reduce the maintenance overhead done e.g. by the structured overlays. By this way, the channel utilisation and the energy consumption could be optimal for all the nodes (sensor, vehicular, fixed, etc).

iv) Security and Trust

One of the emerging issues in the overlay networks is the trust between peers. As this kind of networks is based on the cooperation between unknown nodes, the lack of trust between them may stop the cooperation of some nodes that want to secure themselves from malicious peers. Therefore, our ANA routing framework should offer mechanisms to detect malicious nodes and prevent them from damaging other peers.

2.2.6 Content-based routing

2.2.6.1 Basic assumption and operation

The primary use of the Internet is content distribution - the delivery of web pages, audio, and video to client applications - yet the Internet was never architected for scalable content delivery. The success of today Internet is largely due to the vast amount of contents available at no cost to users. Internet traffic measurements have shown that content access is the dominant service in today Internet. As the number of users in the Internet increases, so is the number and diversity of content.

However, today’s networking protocols and devices do not meet the needs of the content related services. Current services on the Internet are limited to those in which a connection is established based on the IP addresses of the machines. The dominant routing protocols in the Internet such as Open Shortest Path First (OSPF) and Border Gateway Protocol (BGP) are capable of routing packets based on IP addresses. However, these protocols have no knowledge of which server (IP address) is suitable for a particular content.

In the present Internet architecture, it turns out that being IP-smart only is not being smart enough. In addition to being IP-Smart, being content smart is quite beneficial in various circumstances.

A content-based network is a communication network that features a new advanced communication model where messages are not given explicit destination addresses, and where the destinations of a message are determined by matching the content of the message against selection predicates declared by nodes. Routing in a content-based network amounts to propagating predicates and the necessary topological information in order to maintain loop-free and possibly minimal forwarding paths for messages.

Content-based communication is a communication service whereby the flow of messages from senders to receivers is driven by the content of the messages, rather than by explicit addresses assigned by senders and attached to the messages. Using a content-based communication service, receivers declare their interests by means of selection predicates, while senders simply publish messages. The service consists of delivering to any and all receivers each message that matches the selection predicates declared by those receivers. In the content-based service model, message content is structured as a set of attribute/value pairs, and a selection predicate is a logical disjunction of conjunctions of elementary constraints over the values of individual attributes.

Various solutions are explored today, based on XML (see Sarvega) or other content descriptor solutions. Open issues are numerous in this framework.

2.2.6.2 Limitations and requirements

i) Resilience

Major CBR (Content Based Routing) algorithms are still suffering from self-recovery when some nodes concerned by content distribution stop for any reason. In fact, in CBR routing algorithms, the receiver is the only one who determines message delivery not the sender. This property forces the concerned users which have not received the distributed content to ask again for it. In the context of the ANA network, we have to design a mechanism that allows automatically redistributing the content to the subscribers that have not received it without asking for it again. Scalability as a function of the number of nodes and forwarding criteria is another important concern.

ii) Latency

The diffusion of the messages to all the network or even to a part of it make the latency of message delivery grow more and more as the number of subscribers and network members grows up. The ANA routing framework should find a solution to remediate to this problem and select the best routes to forward messages to the concerned nodes.

iii) Overhead

As the CBR are based on event notification and on mechanisms of advertising the content held by nodes, we have to think how to reduce the overhead of these events in the autonomic network that aims to minimize the transmitted control-messages (advertising, event notification about the content, etc).

2.3 Conceptual Analysis of Routing Requirements

As already mentioned, the aim of this deliverable is not to simply review routing protocols per se but rather to highlight the challenges faced by Internet routing and the extreme variety of networks that compose today’s Internet with regard to scalability, connectivity, resource admission, management complexity, security and resilience, performance, etc. The goal is to better understand how we may successfully evolve from the current one-size-fits-all routing model (i.e., IP everywhere) towards a more dynamic and adaptable routing model that can satisfy both local networking constraints and global reachability. This suggests the main landmark of an autonomic architecture.

Furthermore apart from the currently deployed routing paradigms, there is a much richer and wider variety of paradigms and architectures in the literature that have not seen the light of realisation and thus cannot provide us with any lessons. This is either because they have not been particularly suited for certain application domains, or because they are still at their infancy, or some other times because we lack the engineering experience yet to implement them efficiently (despite the fact that we realise their usefulness – e.g. Anycast).

This section considers the problem of routing from a more theoretical (as opposed to empirical) and high level point of view. It discusses and briefly summarises the multiplicity and variety of routing approaches proposed in the literature and classifies them based on different aspects that can serve the deduction of requirements that our proposed ANA routing framework needs to fulfil. The goal is also to capture the essence of diversity and extensibility that an autonomic architecture requires.

Although not exhaustive, this section tries to capture and exemplify the multiplicity and variability of routing approaches that ANA should be able to generically accommodate. The classification that follows is driven and based on different aspects that a network infrastructure can be optimised in order to serve the purpose it is designed for.

2.3.1 Network Structure

Looking at the problem of routing from the network topology perspective enables us to understand the implicit role of address organisation and allocation in the computation of routing paths and the signalling traffic overhead aspects such as address aggregation in hierarchical addressing schemes typically lead to economy of disseminated information, cheaper path computation due to implicit inference of paths, and thus faster convergence time. This is particularly important for scalability. On the other hand when the address hierarchy does not map well to the physical topology, the computation of routing graph leads to incomplete path acquisition, and often existing alternative physical paths are not projected in the routing graph. This renders the routing topology vulnerable to connectivity fluctuations. In such cases a more “flat”/unstructured addressing scheme or one that provides a less deep hierarchy, might be more efficient at the cost of disseminating more information or reducing scalability.

- Structured
- Unstructured

2.3.2 Communication peering and communication direction

The number and choice of participants in a communication exchange presents a challenge for any routing architecture from the perspective of computing paths as well as the amount and type of information that needs to be maintained. For example the current internet is tailored towards unicast routing and any certain routing path assumes two communicating entities at any moment in time. Facilitating multicasting in this architecture is “camouflaged” in unicast addressing while on the nodes responsible for interpreting what seems to be a single peer-to-peer path, to a tree graph of interconnected paths face a number of challenges in terms of computation, scalability, state maintained, etc. And even when this seems possible in a fixed wired infrastructure, it is not at all an obvious approach in mobile networks or wireless environments.

Another challenge is the implementation of broadcasting in non-broadcast networks. While simplicity rules in network environments where routing is carried out at the RF level (such as the TV broadcast network), the solution is not so simple to tackle in topologies where broadcast resolves to unicasting every single participant at a lower level (MAC or physical). In such cases distinct routing topologies may be needed to be maintained independently at more than one layers (in order to hide the complexity), which results in more signalling information (often duplicated) and locality becomes an issue, requiring more “intelligent” computations in order to avoid crossing multiple times the same physical links.

A second aspect is the direction of communication. Routing consistency is better ensured when bidirectional routing paths are assumed. Stated alternatively, routing in face of asymmetry is always a challenge to achieve in an error-free way, let alone the additional state information required and various mechanisms that safeguard against cyclic paths and routing loops. In other cases routing asymmetry may be inevitable due to the type of communication (e.g. in multicasting, or broadcasting) or the physical connectivity (e.g. satellite links).

Careful consideration of the type of communication will enable performance effective versus scalable and possibly resilient routing.

- One-one (unicast)
- One-many (multicast)
- One-any (anycast)
- One-all (broadcast)
- Many-One (concast, network coding)
- Many-Many (multicast)

2.3.3 Visibility

One routing mechanism that is often used to improve efficiency as well as provide some sense of security is indirection. In contrast to proactively maintaining routing state that renders communicating peer candidates visible to each other, in the case of indirection visibility of communicating peers is possible only during the communication per se. This simplifies support for mobility, as well as providing a powerful mechanism of transparently introducing functionality in the network. From a security point of view this obviously provides protection for the communicating ends. However on the other hand it often requires an increased amount of state maintained in the network (NAT, Onion) while at the same time in the current Internet model it often violates fundamental architectural principles (NAT). Under careful design whenever needed indirection should be possible to achieve and facilitate as an integral part of a routing architecture rather than as a hack or overlay mechanism. This is actually expected to reduce the required routing state disseminated and maintained inside the network as routing paths will be short lived, whilst at the same time enhancing functionality, improving scalability and security.

- E2E : “See” directly address, (e.g. UDP) vs. “channel” (e.g. TCP)
- Indirection : Partial indirection (path establishment vs. complete communication), temporal indirection vs. spatial indirection

2.3.4 Medium

The type of medium is not directly related to the routing approach deployed. However as it imposes some operational restrictions and upper bounds to the available resources, it is indirectly responsible for the selection of a routing approach over another. For example where wired infrastructure is available and being more robust provides a more sustainable environment for routing protocols that exchange a large amount of state information. At the same time a wireless environment where node mobility is possible a lightweight routing solution is better suited. Other resource limiting factors that need to be taken into account when deciding on a routing scheme, are the addressing scheme employed, the infrastructure capacity and transmission delays, the node computational resources, and the routing path asymmetries and persistence.

- Wired
- Wireless

2.3.5 Connectivity

This is apparently one of the most fundamental and directly influential factors for the selection of the routing approach as it is directly related with the convergence time of routing protocols as well as with the persistence and validity of the computed paths. Therefore there is a significant tradeoff to consider when deciding on a routing scheme that converges fast or accounts for multiple alternative best paths given various metric.

Getting a bit deeper into the challenge faced here one can distinguish between spatial and temporal persistence of connectivity. Spatial refers to the resilience of a connection or path w.r.t. to the movement of a node within a topology (in the face of node or network mobility). On the other hand by temporal we imply the persistence of a connection in time w.r.t. to the availability of a node in time as in episodic environments or environments where power saving enforces some nodes to be available temporarily, or on-demand. The selection of a routing scheme should be able to take such parameters into account.

- Fixed (spatially and temporally persistent connectivity)
- Mobile (ideally spatially non-persistent connectivity: Mobile Nodes vs Mobile Networks, Macro-mobility vs Micro-mobility, Movement patterns and algorithm convergence, Persistence of paths, Persistence of connections
- Episodic (ideally temporally non-persistent connectivity)

2.3.6 Path Diversity

In accordance with the aspects that were mentioned previously there may be need maintaining a set of multiple routing paths for redundancy and resilience of the infrastructure. However this comes at the cost of complexity (multi- metric scheduling) and increased resource requirements both computationally and storage-wise. This is another aspect that needs to be taken into account as it is a trade off for low power devices. Also the addressing scheme employed may have an impact on the storage requirements for maintaining the routing information as well as the amount of information disseminated.

- Single path
- Multi-path

2.3.7 Path-Acquisition

Another interesting aspect of deciding on a routing scheme over another is the acquisition of routing information and computation of routing paths. Obviously in large fixed infrastructures proactive computation of routing information which is permissible due to the paths persistence leads to latency improvements in the data delivery, as well as bursty interference phenomena of the routing information dissemination with the user traffic. On the other hand in temporally and spatially rapidly changing topologies a more re-active approach might seem more attractive and effective. Finally another paradigm which seems attractive in some cases despite the computational overhead involved is the on-the-way, per packet computation of the routing path advocated by active networks, active agents, mobile agents, and internet worms. In all these cases, flexibility, efficiency and performance with regard to the problem at hand, are the important factors to take into account before selecting one approach over another.

- Re-active

- Pro-active
- Established en-route (Active capsules)

2.3.8 Path selection

The way that information travels across the network between end nodes also imposes some requirements on the candidate routing scheme. There are three main approaches today for routing information in a packet switched network. The connectionless approach where packets are switched from hop to hop, involving a forwarding decision on every hop based on locally maintained routing information. The connection-oriented approach, which involves the establishment of a fixed route across the network at session initiation time at the cost of additional minimal state that “wires” an input interface with an output interface, but at the benefit of making faster decisions. Finally there is the source routing approach where the intermediate hops need maintain no routing information at all and the decision for a certain path is left to the end peers. Depending on the which approach is adopted there may be more or less routing information dissemination needed as well as path computation and storage requirements for the intermediate nodes.

- Connectionless
- Connection-oriented : State, virtual vs. physical
- Source routing

2.3.9 Routing Metrics/Costs

Currently the vast majority of routing schemes employed in service networks compute routing paths and make routing decisions based solely on two metrics, one being hop count and the other policy restrictions. Yet, the diversity of application requirements as well as the range of technologies offering different performance characteristics (latency, capacity, throughput, etc) call for routing decisions based on a variety of metrics. Although some routing protocols account for a multiplicity of metrics such as OSPF, however these are not widely used and instead manually set error-prone policy restrictions are enforced to compensate for the inefficiency. In an autonomic environment where the manual intervention needs to be minimised (if not eliminated) selecting a routing scheme for a given environment and application domain, should rely on the availability of the different metric driven requirements.

To that end the availability of mechanisms that will provide the information for metric estimation is also a crucial factor for selecting the appropriate routing scheme.

- Delay/Latency
- Robustness (Errors, connection persistence)
- Hop count
- Capacity
- Traffic use

- Security

From the aforementioned classification it draws that selecting a routing scheme over another is definitely not an easy task and often a non-deterministic one as there are many trade-offs to consider. One has to make a compromise based on a set of policies, requirements, and factors that depend on the deployment environment and resource availability. To add to the complexity sometimes several of the factors affecting the correct choice of a routing scheme may not be quantifiable or measurable.

A summary of requirements that can drive the selection of a routing scheme over another is the following:

- Network Structure (Structured, Unstructured)
- Communication peering (One-one, One-many, One-any, One-all, Many-One, Many-Many)
- Visibility (E2E, Indirection)
- Medium (Wired, Wireless)
- Connectivity (Fixed, Mobile, Episodic)
- Path Diversity (Single path, Multi-path)
- Path-Acquisition (Re-active, Pro-active, Established en-route)
- Path selection (Connectionless, Connection-oriented, Source routing)
- Routing Metrics/Costs (Delay/Latency, Robustness, Hop count, Capacity, Traffic use, Security, etc)

However, although probably optimality is not possible to achieve, sub-optimality should not be. As a result we expect that the proposed framework is capable of demonstrating sufficient flexibility and extensibility for evaluating robustness against such sub-optimality, whenever quantifiable metrics are available.

2.4 ANA Routing Framework

The analysis above showed that today's networking landscape includes many different types of networks, which have fairly diverse settings and requirements. From this, it is clear that there exists no "one-size-fits-all" routing scheme that can satisfy all the different network types.

As a result, ANA does not aim to provide one single, "magic" routing scheme, but rather aims to provide a routing framework that allows flexible customization according to the type of network and its needs.

The following section will first provide a brief overview of the basic ANA abstractions and the communications paradigms, from which some high-level assumptions and goals for the ANA routing framework will be derived.

2.4.1 Basic Concepts and Goals

The first version of the ANA architecture is defined in the ANA Blueprint document [91]. This section briefly recaps the fundamental abstractions that are necessary to understand the concepts and application of the ANA routing framework.

The following basic ANA abstractions are relevant in this context:

- Functional Blocks (FBs): FBs define the processing elements or functions of an ANA node. The FBs are essentially the entities that generate, consume, process and forward information in an ANA node and network. FBs are executed on an ANA node, where they provide necessary functionality for communication in an ANA context, such as a protocol stack or a routing function. FBs can have zero or more input and output interfaces. Routing within ANA takes places among FBs that are connected (both within a node and across nodes) and thus form an ANA network.

- Information Channels (ICs): ICs define the channel or medium over which communication between functional blocks takes place. ICs can be either of physical or logical nature. Examples of physical ICs are a cable or memory. A logical (or virtual) IC can represent a chain of “underlying” FBs and ICs.

- Information Dispatch Points (IDPs): IDPs are the “connectors” between FBs and ICs. In other words, IDPs are the entities that “bind” ICs to FBs or directly connect two ICs. The main advantage of IDPs is that they enable dynamic (re-)binding of FBs and ICs in a way that is transparent to the FBs and ICs. That is, the IDPs decouple the direct binding and hence allow transparent replacement of one of the elements – the other element still uses the same IDPs and does therefore not need to be aware of the change. Since IDPs also provide functionality that is related to the information processing, it could be argued that IDPs are just a special type of FBs. Nevertheless, because it is considered architecturally important to allow transparent (re-)binding of ANA elements, IDPs are considered one of the key abstractions. In contrast to FBs, IDPs are however limited to perform only data forwarding operations (based on the bindings). Any other type of information processing is done by FBs. As a consequence, IDPs are not involved in the routing process.

- Compartments: As one of the main abstractions of ANA, compartments define the operational rules and administrative policies for a given communication context. The boundaries of a communication context, and hence the compartment boundaries, are based on technological and/or administrative boundaries (for example, a compartment can be defined based on a certain protocol and/or addressing space, but also based on policy domains). ANA anticipates that many compartments co-exist and that compartments are able to interwork on various layers. More precisely, the compartment defines the “recipe” that specifies how the compartment operates; e.g. it defines how to join a compartment, who can join, and how the naming, addressing and routing is handled. Moreover, the compartment defines how communication among its members is achieved (e.g., what protocol is used, what naming and/or addressing schemes are used, how routing is achieved). It is important to note here that compartments have full autonomy on how to handle compartment internals communication – i.e. there are no global invariants that have to be implemented by all compartments. The communication end-points or members of a compartment are represented in the ANA architecture through functional blocks (or composed functional blocks – i.e. sets of one or more functional blocks hosted on a single ANA node) that constitute the “compartment stack” – the functionality required to interact with the compartment.

As indicated before, ANA targets to be a very generic network architecture that is able to accommodate any type of networks. The compartment abstraction allows ANA to handle different types of networks and communication schemes. It allows different types of communication schemes (protocols, address, routing, etc.) to co-exist and interwork (given that the appropriate interworking functions are provided). For example, a single ANA node can be part of many different types of compartments at the same time (represented by one or more FBs). Communication across compartment boundaries is possible through FBs (or composed FBs) that are part of several compartments and thus can act as intermediaries (translators). One of the most challenging aspects in networking is to support end-to-end communication across heterogeneous networks. Compartments have been specifically designed to address this requirement. Hence, ANA allows flexible and dynamic layering of compartments – on top of each other, which allows e.g. the dynamic introduction of a common communication layer for those entities that want to directly communicate with each other.

2.4.1.1 Implications for the ANA Routing Framework

Routing in the ANA world can be split into the problem of intra- and inter-compartment routing. However, the fact that inter-compartment routing can be solved through layering of a compartment on top of a set of underlying compartments, which then handles inter-compartment routing based on another, common identifier, allows handling of the inter-compartment routing problem by large as an intra-compartment issue.

The fact that ANA supports co-existence of different types of compartments allows the application of different type of routing schemes for different scenarios. Moreover, the compartment concept allows customising or tailoring of the routing scheme to the specific needs of the network type that is handled by the compartment.

Routing in ANA compartments is handled by the routing function, which is responsible to select the communication paths through the compartment topology, which is defined by the member FBs and the ICs that link/connect the FBs (or IDPs) together.

In an ANA context, FBs can directly interact with the compartment to obtain routing information. They can, for example, ask the compartment, how to send information to a certain destination, or how to reach a resource with certain attributes. The routing function will provide the requested information, namely how to transmit (send or forward) information towards a given destination. This information will include some identifier that will tell the FB how to handle (forward) the data. This identifier could stand for an interface ID, an IP style address, a HIP style identifier, a label that denotes a dedicated path, etc.

It is important to note that the compartment has full autonomy on how to handle addressing and routing. It is up to the compartment to choose if it wants to reveal the insides of routing to its members. For example, the compartment may choose to use short-lived labels that indicate only how to handle/forward data at this point in time. This way, the compartment reveals no information on how routing is handled by the compartment.

2.4.2 Routing Model

Providing a single routing protocol that will encompass the needs of new and future applications is not feasible. The ANA routing model is designed around a component-based approach. Such an approach helps in meeting various routing schemes and different application requirements. The advantage of a routing component-based architecture is simplifying the design of routing protocols that can target some specific applications and different environment parameters. Instead of reinventing the wheel by building protocols from scratch, the ANA framework gives the ability to combine one, some or, all routing components. A particular combination of the routing components determines the behaviour of the expected routing protocol. While each component can be implemented using various techniques (technical approaches), standardized interfaces need to be well specified to allow composition. The ANA routing framework helps in better understanding the interaction between the selected routing components.

The four identified fundamental components of the routing framework are: addressing/positioning, network topology discovery (dissemination, discovery), route/path determination (selection/computation), and forwarding. As mentioned earlier in section 2.1, we explicitly include in the routing framework components that are closely coupled with routing but which were traditionally kept apart, namely addressing/positioning and forwarding. The objective is to accommodate emerging routing paradigms which do not always draw a clear line between these components.

This section gives a brief overview of these four components, describes their role in the overall routing process and gives some examples where and how they are used. The goal of this section is to identify the functional blocks a generic routing framework must provide in order to better understand and specify the ANA's routing issues. Different possibilities for the combinations are summarized in Fig. 1 (note that we do not consider naming as a fundamental building block). As we will see in Section 2.4.3., a particular combination and the importance given to each one of the building blocks determine together the behavior of the routing scheme.

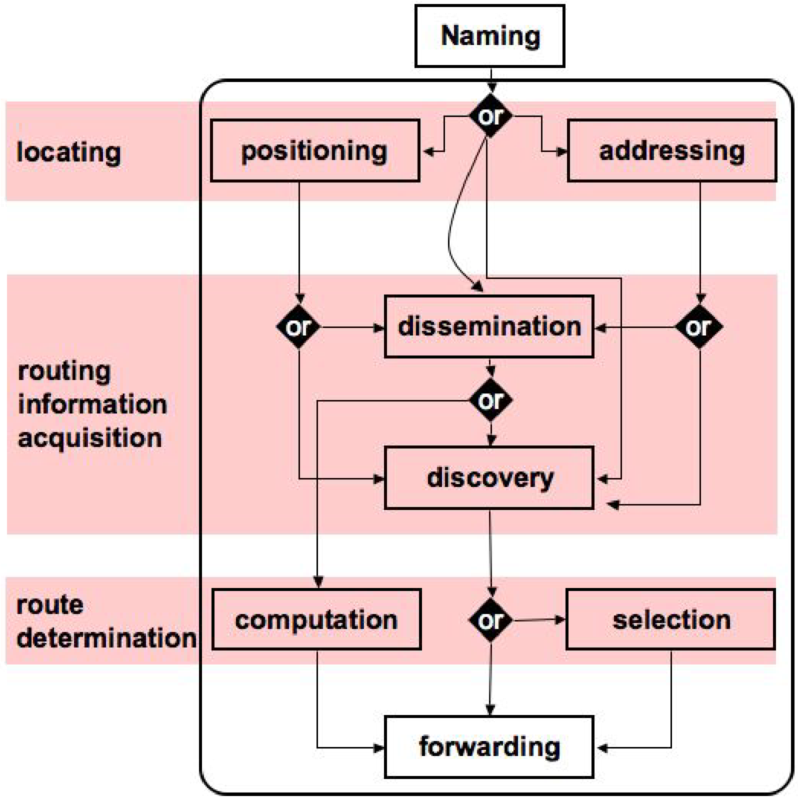

Figure 1: Flowchart of possible combinations of the four ANA routing components

2.4.2.1 Addressing/positioning

The address of a node gives its topological location. It is used as a locator, i.e. a piece of information denoting the actual location of a destination inside a network, independent of whether the address has topological meaning or not. Theoretically, any piece of information can be used as an address, or in other words, as soon as the routing protocol utilizes some identifier as routing tag it is used as an address. For a forwarding process, a destination address or parts of it can be seen as a primary key into the forwarding table with which it can extract relevant forwarding information, such as next hop or outgoing interface.

The way addresses are assigned to nodes is a fundamental issue and has a direct impact on the routing schemes. Addressing schemes can be classified in two main groups: topology-based and position-based ones. In topology-based addressing schemes, either a flat or a hierarchical strategy can be applied. In the flat addressing scheme, nodes choose an address (almost) independently of the other nodes. In this case, some collision-avoidance mechanism must be implemented to guarantee that two nodes do not use the same address. Such a mechanism is inherently available in hierarchical addressing schemes, where the main objective is scalability. Nevertheless, hierarchy implies structure, and a number of drawbacks may arise. For instance, in the case of mobile networks, the use of hierarchical addressing structures may be prohibitive because of the associated maintenance cost.

A possible alternative to the topology-based approach is to define an absolute and collision-free addressing mechanism through a position-based, geographic, scheme. In this case, the node's address is equivalent to the node's geographic position.

There are some special cases of addressing schemes. One of them is clustering, where nodes are organized in groups based on some "proximity" metric. Clustering falls within the class of hierarchical addressing schemes, but presents some flexibility in defining the hierarchy. More specifically, the proximity metric may assume different meanings such as the geographic location of nodes, the connectivity among them, or some semantics that governs their relationships. Another class of addressing systems that is gaining popularity, mainly due to the increasing demand for sensor networks, is addressing by attributes. In this approach, the source does not send data to an individual node, but to any node (possibly many) that satisfies some condition specified in the message; it is up to each node to decide if it is a destination or not. Searches in file sharing peer-to-peer systems, for instance, follow these principles, where the attribute is having a copy of the required file. Many applications of sensor networks also need to use attribute-based addressing. An example is a message addressed to all sensors whose recent measures are above a threshold.

The ANA routing architecture does not restrict routing to one of the wide possible range of addressing alternatives. Addressing flexibility is very important for the overall goals of ANA. Challenges the ANA routing architecture needs to address include interoperability across addressing schemes, flexible and efficient support for renumbering and address lookup.

2.4.2.2 Topology Discovery

In order to find a route, nodes need to acquire network topology information. It can be done on a periodic/event basis or on a demand basis when routes are for immediate use. In the time driven approaches, all nodes share their knowledge of the network topology. In the demand driven approaches, the only nodes that have data to send solicit other nodes to discover routes as they are required. We refer to the process of spreading topology information as dissemination. The process of finding the address/location of a destination and obtaining a route to this destination is referred as discovery.

2.4.2.2.1 Dissemination

Dissemination is the process through which nodes let other nodes know about their address/location and spread their knowledge of the network topology such as path information, next hops, link weights etc. To guarantee convergence and efficiency, information that has some impact on routing decisions is propagated into the network. Such information includes any changes of the network topology due to churn for example, changes in costs associated with links or administrative reasons such as policy changes. As with addressing, dissemination is a functional block that is very different from implementation to implementation and does not have to be present in each and every routing protocol. Dissemination represents the “push” way of sending up-to-date routing information. The most widespread routing protocols, such as the ones used in the Internet, make use of route dissemination as it guarantees available routing information to be accessible when needed. Especially for static networking environments, route dissemination is an effective way to build a routing mechanism as only changes trigger updates. The details of how and which routing information is disseminated depends on many factors such as the nature of the network or the address structure.

The dissemination degree can range from zero to full. Zero dissemination means that nodes keep their knowledge of the network topology information like their own address for themselves. In that case, the discovery component is required: nodes have to explore the network in order to discover the routes immediately needed for effective use. Full dissemination means that all nodes are potentially interested in communicating with the rest of the network as all nodes are seen as potential destinations. There are two traditional ways of disseminating topology information: the distance-vector approach and the link-state approach. Both are table-based in the sense that each node maintains a table that encodes the network topology. In the distance-vector approach, each node updates its neighbours with the best-known distance to other nodes. The link-state requires each node to inform all other nodes of its link cost to every neighbour.

ANA does not limit protocol implementations to a small set of dissemination techniques. The routing architecture is flexible enough to accommodate arbitrary dissemination techniques. What the routing architecture needs to provide is a framework to efficiently implement route dissemination techniques across all layers. Topological information can be beneficial to many layers of a protocol stack. In case many routing protocols run simultaneously, efforts can be coordinated using the framework. A good example would be overlay and network layer routing. In principle the same “thing” is done twice at two layers. Coordination e.g. though cross-layering can greatly increase the overall efficiency of routing mechanisms in use.

2.4.2.2.2 Discovery

Discovery is the process by which a node that has data to send demands other nodes to discover routes to the desired destinations. In contrast to dissemination, discovery is the “pull” way to obtain routing information. It is a well known technique employed in dynamic networking scenarios such as ad hoc networks, where dissemination can be very expensive due to frequent topological changes. Discovery and dissemination are or not mutually exclusive processes. Combination of both approaches is possible. Discovery can be implemented in various ways as well. Central or distributed routing information repositories are one implementation or mechanisms based on flooding are another option for path discovery.

When a full dissemination strategy is used, discovery component is unnecessary, since each node already knows a route to all the other nodes in the network. Some route discovery mechanisms are based on source-initiated flooding. Route request messages are flooded through the network until a node responds with a valid path to the destination. The relative cost of route discovery in such mechanisms tends to be very expensive. Some approaches have been proposed to reduce the broadcast overhead. The expanding ring-search technique increases gradually the area over which the route requests can propagate until the destination node is found.

ANA’s routing framework can support discovery processes and it can help to coordinate and fine-tune the routing information gathering in order to achieve optimal performance.

2.4.2.3 Route determination

Once the routing information is available, the next step is the determination of optimal routes through which information will be forwarded towards the destination(s). The route determination component is the heart of any routing protocol. If only a single route is known or if only flooding is employed this process has no significance. For any more sophisticated routing protocol this process determines the forwarding efficiency. In the routing schemes that instantiate the discovery component, nodes may learn about multiple paths for a given destination. Those routing schemes can support multipath routing by setting up more than one route between each pair of nodes. When single path routing is used, the best or optimal route is chosen among all the discovered paths.

When a full dissemination strategy is used, each node has knowledge of the entire network topology. In that case, a route computation algorithm is used to compute the minimal cost path from a node to all potential destinations. The route computation algorithm depends on how routing information is disseminated. When the entire topology information is disseminated by using the distance-vector approach, nodes determine the routes to all the other nodes in the network by means of the Bellman-Ford algorithm. When applying the link-state approach, the algorithm used is the Dijkstra’s algorithm.

Route computation algorithms can be much more complicated by also incorporating quality of service considerations, security aspects or traffic engineering just to name a few. In addition to purely technical influences, there can also be administrative, political and business-related aspects that have an impact on the path selection process. Commonly those can be expressed with policies, comparable to what BGP is doing in the Internet. Again, the ANA routing architecture imposes no limitation on the actual path selection process. Architectural support can come from mechanisms that allow a flexible combination of the various mechanisms that influence the path selection process.

2.4.2.4 Forwarding

Forwarding is the ultimate process through which information is effectively transmitted. Forwarding decisions are based on some criteria and may require or not route information. Forwarding is used to refer to the process of relaying transit information from (typically one) input (e.g. Network Interface) to at least one output. The performance of the forwarding mechanism (e.g., path length, cost, delay, energy consumption) closely depends on the previous three components. For instance, messages can be forwarded following a hierarchy given by the addressing scheme or a geographic path computed through Euclidean distances between nodes. In geographic routing schemes, forwarding decision is based on the (geographic) position of the destination and the position of the neighbors. Forwarding can be predefined (source routing) or based on a hop-by-hop basis. Nodes select the next hop node towards the destination by consulting a routing table or the packet header based on a predetermined policy.

2.4.3 Analyzing some existing approaches

The ANA routing architecture is the framework in which the above-mentioned components can interact in a structured and meaningful way. Having such an approach greatly reduces the complexity of routing protocol design. Existing components can be arranged in arbitrary ways to make up new protocols tailored towards a target scenario. The way they interact can be defined using the architecture and components can be parameterized according to the requirements of the network. This flexible framework allows balancing the routing efficiency, flexibility and expressiveness in an optimal way for a network's purpose and according to its properties. To illustrate the above, using the modular component-based framework ANA provides, we analyze some existing routing approaches in terms of the routing components. We do not intend here to fully survey existing routing techniques, but to present some approaches from a designer's viewpoint.

2.4.3.1 Reactive (on-demand) approaches

Reactive protocols (e.g. DSR[67] , AODV[54]) establish paths on an on-demand basis. Nodes that have data for immediate delivery initiate a network exploration to discover routes as they are required. Active sources with no routes to destination in their local cache flood the network with route requests. On receiving a route request packet, an intermediate node appends itself to the source route in the header packet (DSR) and may want to store a temporary route state in their local cache if such a state does not already exist. By this mean, intermediate nodes discover in a passive manner, routes to the source that initiated the discovery process. Route requests propagate until they reach an intermediate node that has recent route information about the destination or until they arrive at the destination. A route reply packet is sent back on the reversed source path contained in the route request packet header (DSR) or by using the routing states created by the route request packets (AODV). As route reply packets travel back to the source, the nodes along the reverse path create an entry for the destination in their route table.

Reactive approaches do not require nodes to have global knowledge of network topology and thus reduce the impact of the dissemination component. Maintaining valid routes in face of topology changes require a considerable traffic overhead.

The discovery phase for a given destination may result in many routes replies received at the source that initiated the network exploration. Those replies may originate from intermediate nodes or from the destination. The latter case allows multi-path routing since the destination replies not only to the first arrived route request packet but also to all subsequent received packets. All discovered routes are cached at the source node which is responsible for selecting the best route according to the routing cost function. Route selection can be also performed by the destination node instead of the source node: the destination selects among all received route requests the best route and sends back only one route reply in response to all received route request packets.

Reactive protocols are basically composed by the discovery (the most important component), route selection, and forwarding routing components.

2.4.3.2 Proactive (table driven) approaches

In proactive routing protocols (e.g., OLSR [53], DSDV [66]), all nodes share their knowledge of network topology. Every node maintains a table with entries to all other nodes in the network. Each single node is seen as a potential destination whether or not there is immediate traffic to be sent. In contrast to the reactive case, proactive protocols do not need the discovery component since dissemination is the component that plays the major role. Routing information is advertised on a periodic basis, so all nodes have an up-to-date view of network topology.

There are two ways of disseminating topological information: the distance-vector approach and the link-state approach. Both are table-based in the sense that each node maintains a table that encodes the network topology (or part of it).

In the distance-vector approach, each node updates its neighbours with the best-known distance to every other node. The routing table information is sent on both a periodic basis and an event driven basis: each node sends routing updates to each of its neighbours in the face of topology changes and use similar routing updated received from its neighbouring nodes to update its own routing table.

In the link-state approach, all nodes maintain a view of the entire network topology. To keep these views consistent, each node is required to periodically inform all other nodes of its link cost to every neighbor by using a protocol such as flooding. As a node receives the link states, it updated its view of the network topology

Both approaches apply a shortest paths algorithm to choose the next hop for each destination. The main components of a proactive protocol are thus the dissemination, route computation (selection) and forwarding ones, with the major role played by the dissemination component.

2.4.4 Impact of routing components

In this section, we give some insights on how each routing component impacts on the expected routing behaviour to better target specific requirements.

2.4.4.1 Mobility

Mobility is a parameter of major importance in the design of a routing protocol; it requires a particularly careful design of the building blocks. Assigning to a node an address that is dependent on the node's neighbourhood (e.g., in a tree-like fashion) is not appropriate since neighbourhood information is more likely to change frequently as mobility increases; it is better to adopt an absolute addressing/positioning solution (e.g., GPS) and put more effort on the discovery algorithm. Indeed, since nodes move, their addresses are likely to have a short valid lifetime. The higher is the mobility the lesser is the usefulness of the dissemination phase, and the greater is the need for an efficient discovery algorithm. Indeed, the effort of maintaining an accurate map of the network may be too costly compared to the level of link usage. In terms of the forwarding block, an efficient, adaptive, solution is also mandatory.

2.4.4.2 Fairness

It is desirable that routing protocols be fair in terms of the overhead generated per node. In our routing model, fairness is affected by all routing components; it depends on the physical distribution of nodes in the topology (which impacts the paths that nodes choose to forward packets and the addressing component for certain routing strategies) and on the dissemination and discovery components. In general, fairness is related to the notion of hierarchy. Flooding-based and on-demand protocols are fair in nature, although more overhead is likely to be incurred around the center of the topology. The basic time-based approach is completely fair since all the nodes have the same view of the network. However, improvements to avoid replicated messages establish somehow a hierarchy of nodes and hence tend to be unfair.

2.4.4.3 Scalability

The goal of supporting routing in large-scale networks is particularly challenging, and design decisions may have a great impact on scalability issues. Scalability is closely related to addressing. It is a common belief that hierarchical addressing schemes are more scalable than flat ones. When deciding which addressing scheme should be adopted, other parameters (i.e., mobility) must also be taken into account. For instance, in highly mobile networks, maintaining a consistent hierarchical addressing scheme may not be scalable; it may be more efficient to use a flat approach instead. Concerning the other building blocks, a trade-off between dissemination and discovery must be found. Flooding-based approaches are clearly not scalable. In static networks, on-demand protocols work well, but do not scale in mobile scenarios. In moderately mobile networks, DHT-based dissemination/discovery solutions show better scaling properties.

2.4.4.4 Responsiveness

This parameter is defined as the delay from the time a node asks for a route to the time it starts sending data. It is clearly tightly related to the respective importance one gives to the dissemination phase. Other approaches may improve their responsiveness with auxiliary tools, such as replication and caching. The price to pay in these cases is an increase of the management cost (i.e., for maintaining consistency). In interactive applications (e.g., conversational and games), responsiveness is definitely an issue. However, in many situations (e.g., delay tolerant networks) it would be preferable to have a light dissemination phase and give priority to the improvement of other aspects such as the forwarding mechanism.

We can see from the discussion above that an individual parameter cannot be considered independently from the rest. We can, however, limit their influence by controlling the composition of the basic routing components. In other words, it all depends on the target application the routing protocols are being designed for; given the wide range of possible scenarios, there is no single solution that fits all requirements.

3 SERVICE DISCOVERY

ANA requires a service discovery mechanism for the efficient utilization of the provided network services. In addition, service discovery mechanisms may be employed by other functionalities in ANA, such as routing, self-organization, monitoring, etc. Existing service discovery mechanisms have typically been designed for other environments and present certain advantages/disadvantages; some of them are presented in this section. The proposed ANA service discovery mechanism is also presented along with a lightweight platform that may be suitable for service discovery in ANA.

3.1 Services and Service Discovery

It is important to define the notion of both *services* and *service discovery* the way they will be considered later in this document.

Services: A definition attempt for services is: *any available resource in the form of a running software program (e.g., FTP) or a device (e.g., a printer) , either located in one specific network node (centralized service) or scattered among several network nodes (distributed services)*. Further refinement is possible since in ANA the focus is primarily on "network services" (both distributed and centralized). A definition for network services can be: *any service requiring communication over the network links*. Considering, for example, FTP, it is definitely a network service according to the previous definition since a file transfer takes place among different machines (i.e., over the network links). If FTP restricted file transfers only for the same machine (e.g., like *cp* in Unix-like environments), then it wouldn't have been considered as a network service. In the sequel the term "services" will almost always appear for "network services".

Note that the previous definition does not restrict services to the application layer. Rather, any software component requiring some communication (e.g., message exchanges) complies with the network service definition. A rather descriptive example is routing that is traditionally placed at the network layer and can be considered as a network service. Self-organization, most of the times part of the data link control layer, may also be considered as a network service (e.g., when clusters are created after communicating with their nearby nodes).

A widely used and well-known category of network services at the application layer are the Web Services. The term Web Services is used for services in the World Wide Web which are based on Web protocols (as is HTTP, XML), so that they can be used from the interested clients, no matter where they come from. The Web Services, also, provide a technology which allows the interaction between the services and the clients by using a common prototype in order to succeed the efficient communication of both ends. A UDDI list is a central point of requests in order of retrieving the available services. The list update takes place, usually statically, from the administrator of the list, after a request from the interested service. More information about UDDI can be found on [15] and [16].

Service discovery: Service discovery can be defined as *the process identifying a node*[1] *(e.g., the location) that hosts a particular service (centralized or distributed)*. The reason why we need a service discovery mechanism is because clients do not have an a priori knowledge of the services existing in the network and most importantly about their locations. So they have to be informed on the availability of the services, to be able to derive their positions and use them. Also, in today's pervasive environments that tend to be highly dynamic, with clients and services appearing and disappearing from the system, it is important to keep the nodes updated about the status of the services they already know and dynamically inform them about newly added ones. To sum up, service discovery enables communication and collaboration entities to provide services to peers and to be aware of and use the available services from peers.

The problem of service discovery (sometimes also referred to as resource discovery) can be also seen as the problem of matching a query for services, described in terms of required characteristics, to a set of services that meet the expressed requirements. When this matching is successful, the node hosting the service is the latter's location; this is the information needed (for most of the times) to conclude a service discovery process.

3.2 Introduction

This part presents some key aspects of existing service discovery architectures, aiming not so much to present a state of the art study but rather to present approaches and components that may be or may not be suitable in ANA. Proxy services in transit networks[2] are also presented to conclude this introductory part.

3.2.1 Existing architectures

3.2.1.1 SLP

The Service Location Protocol (SLP) is a protocol designed to simplify the discovery and use of network resources such as printers, Web servers, fax machines, video cameras, file systems, etc. SLP matches the needs of users to the available services in order to determine the appropriate one for each request. It takes care of the service advertisement by employing specific service agents. It also organizes services into directories managed by corresponding agents. Service information is obtained without prior configuration. Services are capable (using SLP) of informing client applications about configuration parameters. SLP is also designed for both interactive and non-interactive use. Applications running at a computer are represented by a user agent understanding the service and resource needs of the application, and each network service is represented by a service agent which makes it available to user agents. Directory agents are considered

[1] Actually what is needed from resource discovery is the piece of information required to communicate with a node where the service is located. This may be node's address, ID, path, etc. depending on the particular routing mechanism.

[2] A transit network is an autonomous domain that does not directly connect to end-systems, but rather solely interconnects other networks.

as service repositories where the clients can look for particular services given particular attributes. A directory agent captures the advertisements from the service agents, collects information from the existing advertisements, and replies on behalf of service agents to user agents when they request a particular service.

The aforementioned mode of operation is a centralized one and not a suitable candidate for service discovery in ANA. However, note that SLP has another mode of operation in which directory agents are not present and messages are flooded into the network in order to determine the particular service. This may be a suitable approach for ANA, provided that special care is taken in order for the flooding procedure to be scalable and efficient. More information about SLP is available at [17], [18].

3.2.1.2 Grid

The Grid is a new class of infrastructure built over the Internet and particularly the World Wide Web. It consists of resources (both hardware and software) shared among participants for primarily scientific purposes.

In order to provide the particular resources, Grids use the notion of "end-systems" (ranging from a wireless PDA to a supercomputer cluster) connected to each other over a rather heterogeneous number of networks. The resources, owned by various administrative organizations, are shared under locally defined policies concerning the sharing permissions. Then, resource discovery takes place in order to provide the requested services.

The resource discovery mechanism in Grids can generate a set of best possible candidates for a given request from a particular client. The idea behind the employed mechanism in Grids is to use distributed databases about the current "status" of the available resources and subsequently serve the resource request queries. If the status databases are organized in a distributed fashion, then the efficiency of the queries will increase significantly, in expense of an increase in the cost of the updates.

Some characteristics of the Grids are listed next:

i) They *scale* in the order of thousands of computers and hundreds of users.

ii) They have *some level of centralization*, e.g. centralized repositories for shared data.

iii) They support *complex operation*, such as execution of a program, read and write access to data, security mechanisms, resource monitoring etc.

iv) There is *stability in resource participation*: with the exception of an occasional failure or an administrative intervention, the resources on the Grid are shared for long, predefined periods of time.

v) They show some level of *homogeneity in usage behavior*, due to incentives that are provided and which also encourage fair sharing.

vi) Finally, in the Grid environment there exists *technical support personnel who is* capable of fixing problems and answering questions.

Non-Uniform Dissemination: A new approach proposed to be used in the Grid environments and which seems to be a suitable approach for ANA, is non-uniform dissemination in order to achieve scalability. The main idea is to inform users in proximity instead of all the users in the Grid about the availability of certain resources, assuming that the further apart located users will not be that interested compared to the close by users. Resource information is disseminated with a frequency that is inversely proportional to the distance from the source, **Error! Reference source not found.**. This results in a significant overhead reduction, compared to uniform information dissemination. In order to achieve this non-uniform dissemination approach, Grids rely on the existence of replicated information repositories that are updated non-uniformly, each one with the resources that are closest to them.

3.2.1.3 The Peer-to-Peer Environment

The Peer-to-Peer (P2P) environment provides a limited specialized functionality compared to the Grids environment (e.g., typically file sharing), but on the other hand it is scalable and dynamic. In particular, it scales in the order of hundreds of thousands of simultaneous computers/users, a unique achievement so far. It employs limited (or no) centralization and most of the times there is an absence of any co-ordination authority. It requires no commitments with respect to resource participation and it is not affected by the heterogeneity of the provided resources. Some more information is also provided in Section 3.2.5, where routing in P2P networks is presented.

Generally, in a P2P environment an overlay network is established and the active peers issue content requests in random intervals in order to find the location of demanded content. In a P2P system, the peers provide the shared content. A peer is simply an application running on a machine, which may be a common personal computer, a handheld device or even a mobile phone. Note that content transfer is not in general part of P2P networking function. The latter are used mostly for content lookup and the actual transfer takes place via additional HTTP connections. There is a certain benefit from this approach since nodes participating in the overlay network (that is set in a P2P environment) do not have to forward the content. Who provides what and which content is available where is not managed by central administration authorities. Only centralized P2P networks employ a central instance as a lookup table, or redirector, which responds to peer requests with a list of peers where the requested content is available. As no knowledge about the topology of the network or the location of the content is available in unstructured P2P networks, these requests have to be flooded through the network. In contrast in structured P2P networks these requests can be routed through the network.

In order to highlight the differences between structured P2P networks and unstructured P2P networks, note that due to the permanent changes of the overlay network, unstructured P2P networks do not put any effort in the management or distribution of the shared content. Content added to the network by new joining nodes is also provided by these nodes, and can only be found on these nodes, as neither the content nor the links pointing to the offered content, are distributed in the network. Thus, in general, requests have to be flooded in the network to reach nodes which can provide information where a specific content is available. The flooding operation has some rules to prevent messages from being forwarded infinitely.

Regarding the structured case, the random distribution of nodes and content may result in an undeterminable location of requested content. Thus the position of content can only be resolved in a local proximity of a node and only by flooding the request to a certain extent. Chord, CAN and PASTRY are currently the most prominent P2P routing approaches which are based on distributed hash tables (DHT). The ID of each node and object may consist of several dimensions, i.e., several IDs for each object. An object in this context may be content or a description of content available in this network. Every node in such a network establishes a preconfigured number of TCP connections to nodes whose keys are the closest in any dimension. If a new node enters the network, it first establishes a connection to a random node, which is already a member of the network, and then is redirected by the peer to those nodes which are the closest to the new node concerning their key. Thus, a certain number of connections must be reconfigured, to guarantee that every node is connected to its closest neighbors. Any content can easily be found, as queries, containing hashed search keywords, must only be routed to that neighbor whose key is the closest to that of the search keywords. Resulting queries can be routed directly to those nodes, which are responsible for the content with the highest probability. Given that in structured P2P networks every query can be resolved independently of the existence of the searched object in the network, as the location of the key of any object is predetermined by the used protocol, flooding of query messages can be completely avoided, as requests can be routed directly to the node which is responsible for the key specified in the request.

To summarize, there exist a number of protocols in the area of P2P networks, whereas most of them are designed for fixed and stable networks. In an ANA environment the dynamicity of the network makes the structured P2P approaches less directly applicable than the unstructured P2P ones. Therefore, although both aforementioned approaches (unstructured and structured) can provide some ideas for designing a service discovery scheme for the ANA environment, the distributed and autonomic nature of the ANA environment is more likely to call for mechanisms related to the unstructured P2P case, due to its dynamicity and lack of centralized authority .

3.2.1.4 Ad Hoc Networks

Much research has been devoted to service discovery in static networks, applied mostly to the Internet. Ad hoc networks have introduced new requirements to service discovery due to the nature and inherent characteristics of these networks. The dynamicity of ad hoc networks creates a number of challenges. In particular, most of the past service discovery protocols cannot work well for the ad hoc networks, because of some characteristics that cannot be accommodated by the very nature of ad hoc networking. In particular, centralized approaches (as mentioned in the discussion for Grids and P2P networks) are not suitable due to frequent (and non predictable) topology changes. Reliable communication is not always the case in ad hoc networks since nodes dwell in a comparably hostile environment (air medium). In addition, not much attention has been paid to particular needs of the environment like the minimization of energy consumption that is essential for the longevity of the network, or the frequent mobility of the users.

Lately, new approaches to service discovery in ad hoc networks have been proposed, e.g., [22], addressing many of the aforementioned problems. These proposals include: enhancements in efficiency (regarding service discoverability and energy consumption) by piggybacking service information into routing layer messages; increasing the bandwidth efficiency in discovering services by using peer-to-peer caching of service advertisements and group-based intelligent forwarding of service requests; designing of a new middleware specifically for service discovery and delivery of device independent services; creation of a backbone of directories constituting a virtual network where each directory performs service discovery in its vicinity (global service discovery is performed by the backbone which is composed of directories that cooperate and are dynamically deployed).

From the aforementioned discussion it is evident that some of the approaches to service discovery in ad hoc networks appear to be in compliance with the ANA requirements for scalability, autonomicity and interoperability among diverse network environments. Therefore, ideas that have been proposed in the area of ad hoc networks appear rather promising for the accomplishment of the ANA objectives and will be carefully considered in the subsequent investigation and proposal for service discovery in ANA.

3.2.2 Proxy Services in a Transit Network

Services may be classified according to their scope. Some services such as printing, local file storage, or even a gateway node to the Internet are for instance local services that only make sense if they are physically close to the requesting user. On the other hand, some global services only make sense if their "providers" are connected together so that they can relay requests that cannot be resolved locally. The Internet Domain Name Service is an example of such a distributed, global service that is expected to be available everywhere.

It is interesting to note that the method employed for service discovery depends strongly on whether the service is local or global. Local services, for instance, have historically been discovered by using some kind of controlled flooding (e.g. expanding ring search) – a technique that should immediately be dismissed when trying to locate the DNS node that knows about a specific name along the hierarchy.

There is a number of situations, however, where the service neither fits a local search, nor a global. Application-level caching and multimedia filtering are examples of such services that could be located in ISPs or even transit networks and for which ring search sounds unappropriate. Unlike the case of local or global discovery, the end-system S (see Figure 2 later in pages 55) that looks for a "proxy" service is planning to use that service when communicating with a destination end-system D whose address has been learnt by an out-of-band method. The distance should be measured based on the shortest path S-D, rather than between the endpoints themselves, in order to limit the path stretch when making a detour through the proxy service.

Among the existing solutions, one can mention "oriented multicast", [1], which suggests a new forwarding mechanism allowing a query message initially flowing from S to D to be duplicated and forwarded in directions orthogonal to the SD path. By "orthogonal", the authors mean that it will be flooded on every interface that does not lead either to S or D, therefore limiting redundancy. Moreover, such duplicates receive a new TTL value (the range) that controls how far from the shortest path the request may go. Unfortunately, the oriented multicast protocol (OMP) does not allow one to control the amount of replies for a given query, except if used in an "expanding oriented search" approach. Moreover, a malicious source could even bomb the destination using a very widely available service and a too broad range.

Alternatively, one could set up an infrastructure that dynamically creates overlay networks interconnecting end-systems and intermediate proxy through tunnels that achieve the desired topology. This assumes that every potential proxy has joined a distributed database that can be searched for proxies nearby a given path. Unfortunately, the existing proposals following this approach (e.g. OPUS, [2], or the X-Bone, [3]) do not come with a truly scalable and completely decentralised method for identifying available proxies in a very large-scale network. Moreover, maintaining the infrastructure (that is, capturing the global network topology), monitoring the available resources and expressing applications' needs in a generic fashion remains a resource-intensive activity, even when hierarchically distributed like in OPUS.

3.3 Potential Use of Service Discovery in ANA

Service discovery is expected to be utilized to some extent in many workpackages and tasks of ANA. For example, monitoring, routing, self-association and self-organization as shown in the following sections.

3.3.1 Monitoring

As mentioned earlier, service discovery is not "yet-another-networking-function". Service discovery should be seen as a key technology used by many other network functions during bootstrap, maintenance, or migration.

Considering ANA Monitoring, there are multiple intersections between service discovery and monitoring. On one hand, monitoring provides information used by the service discovery functional block during lookup; at the same time however, ANA monitoring may also use service discovery functions, e.g., to select or determine the location of monitoring sensors.

In the following, we illustrate two concrete examples how monitoring uses service discovery functionality.

3.3.1.1 Discovering peer monitoring nodes

In a self-managing network, such as an ANA network, decisions to trigger monitoring at a particular point(s) in the network, are taken by some decision makers (functional blocks like protocols, applications, etc.), which may be distributed in nature. To coordinate and orchestrate a monitoring task, these distributed nodes have to know and be aware of each other. Service discovery is hence the bootstrap mechanism used to trigger and initiate the coordinated monitoring task.

3.3.1.2 Sensor/Monitoring Agent Placement

A second obvious interaction between service discovery and monitoring is sensor/monitoring agent placement. Indeed, for this specific task, the focus is more on the service placement aspect of the comprehensive service discovery approach (to be adopted in ANA, see later), than the search for the search aspect, as the objective is to find the best location for the sensor/monitoring agent.

For some monitoring tasks (for example monitoring for traffic anomaly detection), the responsible functional block has to determine where the specific monitoring tasks are executed. For anomaly detection, it is of particular importance to know where the traffic data is collected and what data is expected in the specific compartment (or network). When the to-be-observed traffic is determined, service discovery functions are used to determine the location of the monitoring nodes. It might be not necessary to distribute traffic monitoring all over the network, but only at strategic locations. Since SD knows about topology and structure of the compartment, the SD block is able to identify the best locations for the sensors.

For more information, please refer to later sections discussing service placement in greater detail.

3.3.2 Routing

As mentioned in Section 2, a service discovery mechanism exists in many routing protocols. Clearly, some routing protocols tend to advertise routes for certain destinations in order for the searching process to be easier and faster. The overhead paid by this pro-active approach is the increase number of messages consuming network resources that are important for the maintenance of updated routing table. Reactive approaches limit the advertising process (in most of the case there is no advertising at all) and start route discovery processes as soon as a request has arrived. Clearly, the response time is greater than under proactive approaches. On the other hand, no messages are forwarded in the network when there are no route requests. In the sequel, in Section 3.4, where the ANA service discovery framework is presented, the similarities with the ANA routing framework presented in Section 2.5 will become obvious.

3.3.3 Self-Association and Self-Organization

Self-association and self-organization (considered in Task 2.4) may be benefited by the considered service discovery approaches investigated in ANA.

Self-association may be seen as an attempt for a node entering a network (e.g., waking up, moving to a new network) to become part of it. Becoming part of a network is essential to be able to use the network resources. Traditionally, the node announces its presence by a well-defined broadcast message containing all relevant information that is required for its acceptance (e.g., the node's ID, information about its provider, the traffic profile etc.). This broadcasted information may be received from nearby nodes (either directly or through multi-hop forwarding) and it will be decided (in a distributed or in a centralized manner depending on the particular implementation) whether the particular association request will be accommodated.

As soon as a node is part of a network (association is successfully completed), it will (most likely) participate in future self-organizing processes that will take place with respect to the internal structure of the network. Note that the association of a new node in a network may be a reason for network restructuring and thus a self-organization process has to be initiated. Self-organization is also needed at the beginning of a network's operation but also during transit states (e.g., departure of a node, a broken link, etc). The reason for self-organization (creation of an internal network structure) is primarily to deal with certain requests in a more efficient manner than when no structure is applied. For example, clusters may be created consisting of a well defined set of nearby nodes (see for example reference [40]) and some of them may be assigned special roles like the role of a clusterhead etc.

Consider the self-association case: the corresponding request of a new node entering the network is possible to be served by the nearest clusterhead and as soon as it is completed the new node becomes also member of the particular cluster. In this trivial example, it is clearly shown that self-association requests involve a small number of network nodes being limited in the nearby cluster while the network is re-organized locally without the need for issuing a global re-organization process.

As it appears from the previous discussion, the objective (most of the times) of a self-association process is to identify a certain node that will be able to answer back. The searching for the particular (association) service has many resemblances with the searching for the service process. The location of the (association) service and the set of nodes being aware of the service location also resemble the service positioning phase and the service advertising service respectively. Furthermore, self-organization is about organizing the network in order for certain tasks to be performed in an efficient manner. However, note that service positioning and service advertising are also processes that "organize" the network in order for searching for the service to take place in an efficient manner. As a result, service discovery may share concepts and techniques with the self-organization and self-association process and vice versa.

3.4 ANA Service Discovery Framework

Services typically are not considered to be part of the networks but rather they are the reason for which networks exist: to enable services even though services themselves may contribute to network's ability to support them (e.g., a routing service will enable the information transport that is necessary as part of a service provision).

Until recently, networks were offering only a limited number of services that were provided by specific network entities designated for and capable of providing them. Usually, the decision on the node that would host the service was not an issue, as it was taken in a centralized manner and the choices were limited anyway. The network infrastructure owner or service provider would properly register the service and the search for the service host would be a rather trivial task due to the centralized nature of the traditional environments.

In more recent, more distributed environments (such as P2P, Grid networks, ad hoc networks, etc) the service discovery problem appeared more intensively. Although distributed and typically multi-owner-based and static, the service discovery problem was focused either on centralized approaches or on service addressing schemes that could facilitate the service discovery through appropriate hashing information. The ANA environment is fairly different from the previous centralized and static networking environments of the past, where a small number of services were created and hosted by certain powerful network entities, typically under a common administration. It is also fairly different that the more recent ones (such as P2P, Grid networks where nodes are assuming pre-described roles) and even ad hoc networks due to the diversity of the considered environments. As it was also presented in previous sections, centralized approaches are present in Grids and centralized P2P networks. Addressing schemes are considered in structured P2P networks that allow for the forwarding of service discovery requests appropriately towards the particular service location. Unstructured P2P as well as ad hoc networks do not consider any centralized or well-structured underlying addressing scheme and therefore the underlying ideas seem suitable for an ANA environment.

An ANA environment is shaped by the ANA nodes. These nodes can be resourceful and autonomic. These attributes make these nodes: (a) powerful enough to ***host possibly any service*** but certainly not all the numerous services that may exist or be requested by the ANA users; (b) powerful enough to run all kinds of applications potentially requiring all kinds of network services to be supported (thus, the ANA users can potentially ***request any service***; (c) powerful enough to ***create network services*** which they may provide to the ANA users themselves or host the service in another ANA node (e.g. due to cost reasons); (d) typically mobile, moving around also the location of the services that they may host, changing the traffic patterns for demands for services they request or they host, ***creating a dynamic environment*** in all aspects; (e) self-managed and self-aware enough to ***take own decisions*** (regarding service creation, hosting, etc) and be concerned about its own utility or well being, whatever that means. Finally, the ANA nodes are expected to be part of large networks spanning large topological or physical distances and including a large number of nodes, thus ***scalability aspects*** become crucial as well.

Consequently, services in an ANA environment can be created *by any node at any time*, *can be hosted by any node*, *and can be requested by any node* which may also be *changing its location and take autonomous decisions*[3]. Thus, the service landscape is fairly dynamic in all aspects, as it is the networking landscape due to the autonomicity of the ANA nodes. As a result, the problem of service discovery in an ANA environment is *very different* than that in previous networking environments and the approach to it should be carefully considered. What appears to be a meaningful approach to service discovery in ANA is articulated in the next section.

Apart from the theoretical work on the various aspects of service discovery, it is important to maintain a realistic view regarding realization aspects. A lightweight platform for service discovery is presented in the section after the next (along with its key

[3] The assumption made is that nodes are normal network equipments and not special devices (e.g., PDAs etc.) that have limited capabilities.

attributes) that may provide valuable reference approaches to implementing service discovery for the ANA environment.

3.4.1 Service Discovery Phases

As already mentioned for the ANA environment, the service creation proliferation, the ability of almost any node to host any service, the autonomicity of the nodes and the dynamicity of the service demand process and network characteristics/structure, create an environment in which service hosting becomes fairly dynamic and particularly so if one aims to keep the service provision cost low. As a (large) number of ANA nodes are capable of hosting a given service and service demands and service provision costs change due to the dynamicity of the environment, an efficient service provision scheme would try to place the service at a node such that the service provision cost is minimized. The service provision cost is shaped by various factors such as the abundance or scarcity of the resources of a host, its distance from the bulk of the demand for it, etc. Consequently, the first phase of the service discovery process will be defined to be that of the service placement. Clearly, this problem will need to be addressed in a scalable, low-complexity and distributed manner.

Notice that as the ANA network dynamics change, the optimal service placement will typically change as well. Consequently, the service placement problem in this case would amount to a service migration one, specifying a migration strategy for the original service host, as the network dynamics change. Finally, depending on the intensity and the geographic disparity of the demand for a given service, a decision for a service replication could be taken, placing copies of the service to a number of hosts, in order to cope with the demand and reduce the overall service provision cost.

Apart from determining an optimal position of a service, knowledge about the existence and position of the service is necessary, in order for the service to be utilized. Centralized schemes (e.g., registration of the service at a particular central agent) are not suitable for an autonomic environment (e.g., they provide for a single point of failure, high delays, increased complexity and are non-scalable). Consequently, a more distributed approach to managing and accessing service provisioning capabilities and locations information should be adopted for an autonomic environment. As a consequence, the service host and/or the potential user of the service would need to do some work in order to enable the discovery of a service by a user.

The delay (or efficiency or even success) in discovering the service location depends strongly on the number of network entities that they themselves are aware of the service location. If only the service node is aware of the existence of the service, then the search for the service initiated by the user will have to reach the service node in order to learn about the existence and location of the service.

Typically, the existence and the location of a service become known to network entities other than the service node itself, in order to expedite and make more efficient the search for the service. These other network entities become aware of the service location through service advertisement that is typically initiated by the service node. That is, the service node (or some other entity on its behalf) informs other network nodes of its location.

Clearly, if the service advertisement is implemented as a flooding process, then there is no need for a search for the service by the user, as all the nodes (theoretically at least) should be aware of the location of the service. In this case, the intensity of the search for service is the minimum and it is theoretically zero (there is no need for such a process at all), while the intensity of the service advertisement process is the maximum (flooding). On the other hand, if there is no advertisement of the service (in which case the intensity of the service advertisement process is minimum and specifically zero), the intensity of the search for the service process is maximized, since the absolute minimum number of network nodes (only the service node itself) are aware of and can provide the service location. Thus, one can clearly observe that there is a trade off between the intensity of the two processes: the higher the intensity of one, the lower the expected required intensity of the other.

In view of the above discussion it is clear that service discovery in ANA can be separated into *three different phases*: service placement, service advertisement and search for the service. While these three phases are obvious, they have not been considered in a holistic manner in past networking environments, as these environments were sufficiently static or centralized and some aspects of the phases were trivial or not an issue. As it should have become clear from the above discussion, the dynamicity of the ANA environment calls for a consideration of all three phases. Their intrinsic characteristics should be well understood and proper design choices be worked out in order to develop a scalable, adaptable and effective approach to service discovery for the ANA environment. Ignoring or poorly managing one phase may make meaningless any effort to optimize the other ones.

In the next subsections, work carried out in the context of the three-phased approach to service discovery is described. Although the work towards this deliverable was officially started in M6 of the project, work on the first phase to service discovery (i.e., service placement) was initiated almost from the beginning of the project. The work carried out is summarized in the next subsection and has lead to conference publications (MedHocNet'06, [31], and INFOCOM'07, [33]). It should be noted that this work is fundamental in the sense that (a) it addresses effectively (see next subsection) the generic and widely applicable facility location problem with numerous potential applications in networking and (b) it can be applied to the ANA environment not only in the context of the first phase to service discovery, but also in other ANA contexts (e.g., placement of monitoring agents). Although substantial effort has been put into this problem and the obtained results are significant, the work will be continued in the next year in order to investigate more the performance of the approach, the properties of the developed algorithms and potential improvements, as well as create a platform where the convergence paths, speed and overall effectiveness of the mechanism will be visualized and be demonstrated.

The second and third phases to service discovery have also been initiated by focusing on the problem of limited (scalable) information dissemination schemes, that are central to both of them (i.e., service advertisement and search for service). The work so far has been investigative and preliminary: on one hand the literature has been carefully reviewed in order to identify approaches that will be potentially effective for the ANA environment and, on the other hand, the creation of a generic simulation environment has been initiated to be utilized for the evaluation of the service advertisement and search for service mechanisms that will be designed for the ANA environment; some basic, limited flooding schemes have already been incorporated in this simulation environment.

3.4.1.1 Service Placement

The problem of service placement has received some attention in the aforementioned traditional networking environment, for example, in the context of content placement and replication in Content Distribution Networks, [14]. This problem is typically addressed by invoking approaches that do not scale with the number of services and network nodes, typically rely on some global information knowledge in order to provide for a solution under given (static) conditions and cannot inherently cope with dynamic environments. As indicated in the next paragraphs, the service generation and provision landscape and the supporting networking infrastructure are changing drastically in a way that the traditional approach to service placement is non-scalable.

The first change has to do with the proliferation, ``miniaturization'' and autonomicity of the services produced by networked nodes. The emerging ``long-tail'' relation between percentage of content (service) produced by a certain content (service) producer reveals the fact that network services proliferate in number and type and that most of these services are ``small'' (i.e., easily produced by small networked nodes). In addition, the technology appears to be mature to consider service personalization and autonomic service composition, which is expected to enhance the autonomicity of the network services, as well as further enhance their ``miniaturization'' and proliferation.

The second change has to do with the proliferation, ``miniaturization'' and autonomicity of the network elements, as well as network users. The term ``miniaturization'' may be used here to capture the fact that the traditionally heavy network elements (routers) are increasingly being supplemented by lighter network elements that are contributed by (until recently) traditional network users; these users are becoming powerful enough to engage in ad hoc networking and contribute to the networking infrastructure. For example, the increasing contribution to the networking infrastructure of numerous small (traditionally) user-nodes is already materializing in last-mile networking and is expected to dominate soon (e.g., home owner based WLAN network service access), increasing the autonomicity of the networking infrastructure, as well as contributing the trend of miniaturized and proliferating network elements. In addition, numerous new small appliances are increasingly being networked, contributing to the proliferation and “miniaturization” of the network users. The increasing proliferation of the network elements and their increasing autonomicity and ownership diversification are the main reasons for which a new network architecture is needed to organize and address the efficiency and complexity issues associated with it, as it is the case in ANA.

One main point from the above discussion is the proliferation of services and network nodes that calls for approaches that scale well with the numbers; ``miniaturization'' contributes to the proliferation, as well as calls for approaches that should be distributed and relatively light. Consequently, the traditional problem of placing relatively few big services to one of the few (powerful) potential service provider facilities (big network elements) is increasingly being transformed into a problem of placing numerous services to one of the numerous potential service providers (network elements and possibly service producers).

As articulated earlier when introducing the three phase approach to service discovery, the effectiveness of a search for a service is affected – among other things – by the service location.

3.4.1.1.1 Facility Location Approaches

Known also as Discrete Location Theory, [23], this part of optimization theory may be divided in three distinct problems: the k-median problem, the Uncapacitated Facility Location (UFL) problem and the Capacitated Facility Location (CFL) problem, presented in the sequel. The k-median Problem is perhaps the most common formulation and the others (UFL and CFL) may be seen as extension cases.

The optimal service position depends on a) the service demands for each node in the network; b) the weight (e.g., the energy consumed for a transmission over the particular link, the delay when a certain packet error rate is required) for every link in the network. Note that knowledge of the link weights implies full knowledge of the network topology. For example, assume a certain network topology and weights assigned to the existing links among nodes, and given the service demands for all nodes in the network. Assume also, that the service is located at a particular node (only one service in the network). Then, the corresponding cost will be the summation of all products for all nodes in the network with respect to their service demands and their distance from the network. A formal definition is provided in Appendix A.I.

It has been proved by Kariv and Hikimi, [24], that the k-median problem is *NP*-hard for the general directed graph. For the case of undirected trees they have also proved that the problem has $O(k^2N^2)$ complexity; Tamir, [25], more recently has reduced this complexity to $O(kN^2)$. There is also considerable work in the area of approximation algorithms and heuristics, [26], [27], [28], [29], that try to reduce further the complexity of the problem. More recent work in the area can be found in [14], [30]. The interested reader is advised to have a look at reference [23] for further past related work.

3.4.1.1.2 Service Migration

In most cases, apart from some heuristic or approximation policies the determination of the optimal service node requires some *global information* of the *network status* (e.g., the network topology and the service demands of all network nodes). In addition to the aforementioned complexity and scalability issues associated with the solution to the k-median problem, such approaches become prohibitive in dynamic environments, like an ANA environment, where a repeated application of the approach and continuous dissemination of global status information would be required, to continuously determine the updated optimal service node location. For such dynamic environments, one could either incur the cost of determining the optimal position continuously at predefined periodic intervals, or implement (typically complex) mechanisms for detecting when the optimal service node position is not valid any more and determine the new optimal service node position. In any case, the cost is heavy and the effectiveness of these approaches questionable (i.e., likelihood to be actually providing the service when requested from the optimal location may not be high).

Since such (traditional) approaches are complex and are based on global information availability, they are non-scalable for networking environments supporting numerous services, service users and network elements, which are also expected to be fairly dynamic. A more reasonable approach to service placement for large, ad hoc and autonomic environments would be through *service migration*, [31]. That is, instead of solving continuously a large optimization problem requiring global information, consider policies for *moving* the service position (one hop/node at a time) based on *local* information, *towards* more effective positions, [32]. Developing service migration policies with good properties is a major challenge, since such policies may be sub-optimal (that is, they never converge to the optimal position), follow a non-monotonically cost decreasing path to the optimal position, etc.

In [31], a simple migration policy is proposed for undirected tree topologies. It is shown analytically that the information available at the *current* service node only is sufficient for determining the direction towards nodes with monotonically decreasing cost provision costs, and eventually, the optimal service node.

The service node needs to simply monitor the *aggregate* amount of data exchanged through its neighbor nodes associated with the particular service, and decide on the service movement based *exclusively* on the information gathered through the monitoring process. An important result shows that this information is *adequate* in order for the service to move towards the optimal service node. It is also proved that under the proposed policy the service is finally moved to the optimal service node and remains there as long as the network status does not change significantly; minor changes typically do not result in a new optimal service node position. When major changes occur, the service eventually moves towards the new optimal service node position.

Based on analytic studies, [31], it is shown that the proposed policy: (a) Moves the service to the optimal position when there are no changes to the network status and service demand profiles (i.e., it is efficient in a static environment); (b) Adapts dynamically to changes to the network status and service demand profiles (i.e., it is efficient in a dynamic environment); (c) It is scalable; (d) It has low complexity (no additional message exchanges). Consequently, the proposed policy presents better characteristics than existing (and static) approaches in the area (e.g., [24], [25]).

According to the proposed policy, [31], it is easy to conclude that every movement of the service results in cost reduction. In addition, the proposed policy is of low complexity in the sense that no message exchanges are required among the nodes in the network for the purpose of implementing the policy (it is exclusively based on information available at the service node).

The proposed Service Migration Policy - as it was presented so far - assumes that the service moves from node to node until it reaches the optimal service position. However, there might be networks that this is not realistic either due to a certain "cost" to install the service to a new node or due to the fact that some nodes may not be suitable for hosting the service (e.g. limited processing power, physical memory, bandwidth, etc.). Given the aforementioned limitations, may be the only feasible approach could be to move the service only once in order to place it at the optimal service node (assuming that the latter is suitable to host the service).

Under the aforementioned constraints, an alternative approach that can be considered is to move not the actual service but a *service monitoring entity* (SME). The role of this SME is to monitor the service request process at the locations visited and apply the proposed

Service Migration Policy (referred to hereafter as the SME Migration Policy) and eventually reach the optimal SME (and service) node location. At the end of this process the SME will notify the original service node of the optimal service node position. At each node, the SME will enable the monitoring process and based on its results, it will determine the next hop. Finally, the service will be placed at the optimal position without being installed in all intermediate nodes.

Service movement can also be accelerated by installing a monitoring process at each node in the network, thus, have available the aggregate service demand associated with each node. When the service (or an entity similar to SME previously discussed but with no monitoring functionality) moves to a new node, is already aware about the aggregate service demands of its previous location.

3.4.1.1.3 Distributed Service Provisioning in Large-Scale Networks

Borrowing some ideas from the service migration policy, [31], a new approach to the solution of the facility location problems was investigated, [33]. The main idea of this work is to provide for a scalable approach to automatic service deployment. Under this context we develop a scheme in which an initial set of service facilities are allowed to *migrate adaptively* to the best network locations, and optionally to increase/decrease in number so as to best service the current demand. Our scheme is based on developing distributed versions of the k-median problem (for the case in which the total number of facilities must remain fixed) and the UFL problem (when additional facilities can be acquired at a price or some of them be closed down).

Both problems are combined under a common framework with the following characteristics: an existing facility gathers the topology of its immediate surrounding area, which is defined by an r*-ball of neighbor nodes* that are within a radius of r hops from the facility. The facility also monitors the demand that it receives from the nodes that have it as closest facility. It keeps an exact representation of demand from within its r-ball, and an approximate representation for all the nodes on the ring of its r-ball (nodes outside the r-ball that receive service from it). In the latter case, the demand of nodes on the “skin” of the r-ball is increased proportionally to account for the aggregate demand that flows in from outside the r-ball through that node. When multiple r-balls intersect, they join to form more complex r-shapes. The observed topology and demand information is then used to re-optimize the current location (and optionally the number of) facilities by solving the k-median problem (or the UFL) problem in the vicinity of the r-shape.

There is a trade-off between scalability and performance. In particular, by reducing the radius r, the amount of topological information that needs to be gathered and processed centrally/ at any point (i.e., at facilities that re-optimize their positions) also decreases. This is a plus for scalability. On the other hand, reducing r harms the overall performance as compared to centralized solutions that consider the entire topological information. This is a minus for performance. We examine this trade-off experimentally using synthetic (Erdos-Renyi, [34], and Barabasi-Albert, [35]) and real (AS-level, [36]) topologies. We show that even for very small radii, e.g., $r = 1$ (i.e., facility migration is allowed only to first-hop neighbors), or $r = 2$ (i.e., facility migration is allowed only up to second-hop neighbors), the performance of the distributed approach tracks closely that of the centralized one. Thus, increasing r much more is not necessary for performance, and might also be infeasible since even for relatively small r, the number of nodes contained in an r-shape increases very fast (owing to the small, typically O(log n), diameter of most networks, including the aforementioned ones).

3.4.1.2 Service Advertisement

The second phase of service discovery is service advertisement. In most of the cases when a service is provided, the advertisement is an essential part in order to become known in the network and start to be used. Clearly, a service is a successful one if users use it. As it was described in the previous section, services are expected to proliferate and will not be possible to place a service in every node in the network due to increased provision cost. Consequently, only a limited number of nodes will typically host a service and all other network nodes should have to look for those service hosting nodes. The latter will be facilitated by advertising the location(s) of the service(s) so that (all) potential users can be aware of the availability of a service and its location and use it accordingly. However, the latter would amount to flooding the network with all service locations which is an expensive way for service advertising and inappropriate when scalability is one of the main concerns (as in ANA).

Although work on the service advertising / search for the service processes is planned for year 2 of the project, some preliminary work has been initiated to prepare for the upcoming studies. To get a sense of the properties, weaknesses, overheads and complementarities of the second and third phases, we started building a simulator implementing various types of flooding: from full flooding, to probabilistic flooding to limited scope flooding. A starting point – which could help gain insight and derive certain bounds – could be the employment of a simple probabilistic flooding algorithm for the service advertising.

A simulator implementing a simple probabilistic flooding has been set up and tested, as a first step or first component of a simulation environment we will likely need to build for the service discovery studies in ANA. This algorithm is distributed and implements the following simple generic steps. Each node receiving a service advertising message for the first time makes an attempt immediately and once only to forward it to each neighbor node (different from the one from which it was received) with a probability p_{fl}. If it is not the first time a node receives the particular advertising message, it simply ignores it. Depending on the value of p_{fl}, the aforementioned algorithm is capable of providing a wide range of dissemination instances. For example, for $p_{fl} = 1$, the aforementioned algorithm results in a full-flooding algorithm. For values of p_{fl} close to zero, it is evident that the number of nodes informed will be close to zero as well.

Preliminary simulation results were obtained for various parameter values corresponding to various types and intensities of flooding; some of these results can be found in Appendix A.II. It can be observed (as expected) that for small values of p_{fl} (corresponding to a limited advertising process) the number of messages exchanged and the percentage of nodes informed remain low (close to zero). For large values of p_{fl} (intense advertisement process) a certain maximum is assumed with respect to the number of messages exchanged and the percentage of nodes that are informed. This maximum is actually determined by the size of the connected sub graph of the network containing the service node.

Between the maximum value and the minimum value (for both the number of messages and the percentage of informed nodes) there is a small period of a sudden increase, as p_{fl} increases, which is indicative of a *phase transition phenomenon*. This observation suggests that there are values of p_{fl} - just after the phase transition - for which the percentage of informed nodes can be close to maximum with comparably small number of exchanged messages; beyond these values, the increase in the percentage of informed nodes is disproportionate (relatively small) to the large increase in the number of transmitted messages.

3.4.1.3 Search for the service

A first simple approach for locating a service is to register all services in a single centralized entity and then retrieve the information whenever it is needed. This simple approach serves well when the number of the services is small and the network size is also small. For a large number of services and large networks with autonomic characteristics (see earlier) it is apparent that such centralized approaches are not suitable.

When a service registration process involving well known service-location information hosting entities is not available - as it is expected to be the case in a dynamic, distributed and autonomic environment, e.g., see ANA – a search for the service process needs to be established. In the extreme case in which only the service providing entities are aware of this capability (and thus its location) it is necessary for the entity in need for a given service to send out flooding messages in the network in order to acquire information about the service by reaching (with probability one for a connected network) the service node.

When some service advertisement is implemented, the number of nodes aware of the service location is increased and the search for it is facilitated. If full flooding is again implemented in search for a service, it is expected that the service discovery delay will be reduced compared to the case of no service advertisement. That is, service advertisement can help reduce the service discovery delay.

Full flooding is a “desperate” or “brute force” approach. It is very costly in number of messages and unrealistic for large networks, although it can guarantee that the service - if available - will always be discovered, provided that the flooding messages can reach all nodes in the network (no network partitioning).

As the ANA environment is bound to have uncertainties in various aspects due to the autonomicity and dynamicity of the environment, probabilistic guarantees should be considered more than deterministic ones. It would be a waste of resources to try to implement policies with deterministic guarantees in an environment that is inherently uncertain in several aspects. Corresponding approaches should be appropriate for ANA and will be considered.

A promising (probabilistic) approach to searching for a service in ANA would be one that is based on Random Walks (RW), [47], [48], [49], [50]. That is one or several agents (service request messages) are sent out to look for a given service. The path that the agent follows can be a pure RW or a modified RW; a modified RW would take into consideration real dissemination constraints, or possible and rough/past service location information, or a super-node structure, etc, to deviate from a pure RW. Clearly, the overhead associated with a RW-based search is expected to be much lower than of full flooding.

A RW-based search could also be supplemented by limited flooding, leading to a hybrid scheme. Or, one could consider generalized search schemes which would include RW and flooding as special modes (among others) and which could adaptively intensify the engagement of a certain search mode, depending on the environment, constraints and requirements.

The service advertisement and search for the service processes can be viewed as dual processes to a certain extent and similar algorithms could be used to implement both. For instance a RW-based approach could also be followed for service advertisement. In addition, the impact of the employed service replication/placement process on the service advertisement/search for processes will need to be considered as well.

In view of the previous, it seems to be necessary to also consider jointly the advertisement and search for processes. This should be done both analytically, as well as through simulations. Along these lines, we have initiated the integration of the search for service process with the advertisement one in a simulator that implements just the probabilistic flooding algorithm. In this case, a node, called the initiator node, starts the searching for the service process. This initiator node is selected by a random function and starts flooding the network with (high) probability $p_{fl,s}$, in order to find either the node that the service is located at, or a node that has information about the location of the service (from the previously completed advertising process). The flooding process is similar to the advertising process, as already mentioned. The searching process terminates successfully if the location of the service is found (or a node keeping this information), otherwise there is a failure.

Various simulation results were obtained and some of them can be found in Appendix A.III. The results show that a high percentage of informed nodes is needed to achieve a high probability of success. For a given intensity of the advertisement process (p_{fl}), it is interesting to observe that almost the same probability of success may be achieved for a range of (low) values of $p_{fl,s}$; by selecting a low value of $p_{fl,s}$ from this range, a low overhead (in number of search messages) of the searching process can be achieved for about the same probability of succeed. This behavior is due to *the phase transition phenomenon*. It seems that a good rule that yields the aforementioned efficiency is to select values for p_{fl} and $p_{fl,s}$ that are slightly higher than the value for which the phase transition phenomenon takes place. Such initial observations show that the processes of service advertisement and search for service should be carefully designed and studies to understand well the involved trade offs and eventually develop effective service discovery schemes for the ANA environment.

3.4.2 A Lightweight Platform for Service Discovery

Service discovery is not only the problem of end-systems. In different places, both machines that offer and that request services could benefit from a more flexible network infrastructure. In this section, we detail a candidate lightweight platform (WASP [4]) developed at University of Liège in order to improve flexibility in the network and help tasks such as service discovery.

Currently, WASP exists as a module for the Linux Kernel and a proof-of-concept implementation is under completion for the IXP network processor hardware, potentially enabling the support of our lightweight platform on middle-range routers with Gigabit interfaces.

3.4.2.1 WASP: World-friendly Active packets for ephemeral State Processing

The two core components of the WASP platforms are the State Store and the WASP packet. The state store is a key-value repository associated with a given network interface. Packets that use that interface may leave or retrieve information from the store using a 64 bit *key*. Through a simple (and safe) bytecode language, WASP packets are capable of collecting, aggregating and exchanging information by manipulating values stored on the routers they cross. Several restrictions are enforced to guarantee network and nodes safety in all circumstances while still allowing enough flexibility.

Beside state store manipulations and arithmetic operations, the programs contained in WASP packet may also request basic information about the current node (such as its address, or interface usage statistic) and about the current packet (e.g. the value of the TTL field or the current destination). Finally, the language has primitives to decide on the packet's fate, i.e. whether it should be forwarded, dropped or returned to its source.

A key idea of WASP is that the state store is an anonymous facility available for all. Using the correct key is the only proof we require from an end-system to make sure it is allowed to read or manipulate information stored under that key. Similarly, as developed in [46], the size of the store and the lifetime of an entry are adjusted to the wire speed so that every packet is equally allowed to create new entries.

We're currently investigating to what extent we can allow modification of packet targets using the WASP platform without messing up routing policies and end-system security. In the case of Service Discovery problems, it would also be interesting to consider how we could build customized (controlled) flooding mechanisms by combining a generic mechanism that relays a message on all interfaces and the processing ability of the platform to selectively drop messages.

Even though WASP has been designed before the first drafts of ANA blueprint, it integrates nicely with the ANA framework. First, all interaction between WASP and the other programs goes through the state store. If any other component of the ANA Playground need to interact with WASP program, it is thus possible to map a part of the address space onto values stored in the key/value repository of the MINMEX core (see the ANA Blueprint document). Moreover, the processing of WASP packets perfectly fits the behaviour of a single functional block that would be chained between packet reception/transmission hooks and a generic "routing" block. The instruction set of WASP could easily be extended so that packets can be handled to different functional block through key resolution in the internal dispatch table.

3.4.2.2 Service Demand Aggregation using WASP

We have seen in section 3.3.1 that correct estimation of the aggregated demand is important if one wishes to have a working service migration mechanism. When a node *y* plans to migrate its service to one of its neighbours, it will need to know the aggregated demand from each of its neighbours . This is a good example where the WASP platform could help, especially when we are interested in the statistics a few hops away (and not only those from immediate neighbours).

Every client that sends a request *r* to *y* will attach a WASP program $count(K_y, \lambda_r)$ to the request packet. When this program is processed by a node, it will first retrieve the key K_y that has been randomly chosen by node *y* for statistics aggregation. Under that key, it will find in the state store the running aggregate demand for node *z* and add the estimated contribution of the current request λ_r.

The result is that the network is now "labelled" with the aggregate values for every intermediate node *z* between *y* and one of its clients. The service can thus process requests without bothering about the individual λ_r carried by the packet. When the servicing node wants to know the running aggregate demand from the different neighbours, it just sends a $get(K_y)$ program to its immediate neighbours. Refining the information to the *i*th hop is as easy as sending the $get(K_y)$ program to the merging points we are interested in.

In the above example, we assumed that clients know at the time they issue the request which machine will process the request and what key is used by that machine to aggregate statistics. When the service discovery mechanism is separated from requests routing, this can be achieved simply by including the key K_y in the service advertisement message, or by constructing K_y through a hash of *y*'s public identity. When this assumption no longer holds, or when we cannot assume that clients are capable of estimating the contribution of each request, we can simply have the $count(K_y, \lambda_r)$ attached by the service node on the *reply* packet.

3.4.2.3 Proxy Service Advertisement/Lookup with WASP

One of the advantages of the *oriented multicast protocol* (OMP) is that it requires no state to be kept in the router, but the price is that every node in the network must support the OMP extension so that the request effectively reaches agents.

An interesting alternative developed in [4] is to consider the case where only a portion of the routers (including edge routers in a domain) play a significant role in the discovery process while the core of the domains can keep working as usual. In this approach a special *state store* allows proxies hosted in a given domain to drop service advertisements in the edge routers of their domain. These advertisements can later be recorded by a connection establishment packet so that the end-systems learn the potential proxies located near their shortest path.

In this scheme, it is assumed that each proxy function is mapped to a flat identifier (e.g. a 64-bit hash) that both the proxy and the end-system know. The enabled edge routers provide an associative array where those identifiers will resolve in a (*addr*(*P*), *cost*(*P*,*X*), *load*) tuple giving the address of the proxy node (*P*), the hop distance between *P* and the router on which the advertisement is stored (*X*), and an estimation of the current load of

P. If two (or more) proxies supporting the same function are installed in a domain, this gives them the opportunity to discover each other and to automatically adjust their advertising ranges so that only the closest or the less loaded one(s) are advertised on the router.

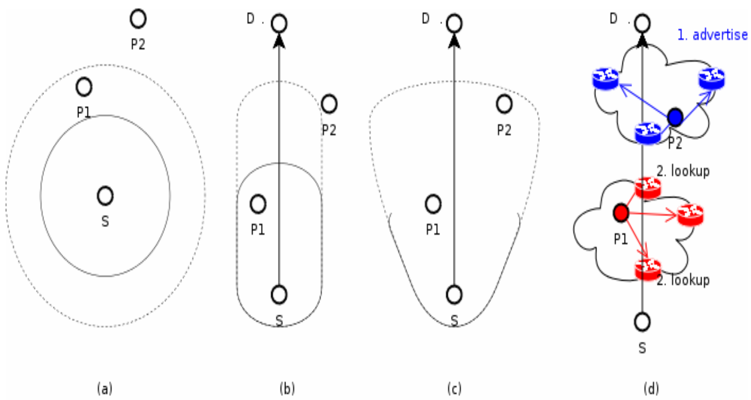

Figure 2: Comparing expanding ring search (a) with two variants of oriented multicast search (b and c), and the domain-wide advertise/lookup mechanism (d).

When an end-system wants to establish a new connection through some proxy providing a specific function *f*, it will send out a packet that records tuples ($addr(P_i)$, $cost(P_i, X_i)$, $cost(S, X_i)$) from information found under *hash*(*f*) in the different (border) routers X_0 ... X_n it crosses on its path and report this information back to the connection initiator *S*. *S* will then have to inspect the recorded list and pick the proxy *P* that will minimise the cost of *S* --> *P* --> *D*.

In order to advertise the service, the proxies will have to locate compatible routers that are capable of storing the advertisements. The most straightforward option (if available) will be to send a single advertisement packet to the "all routers" multicast group. If the topology built for a link-state based routing table is available locally, it can be enough to learn the addresses of the domain's edge routers, [5], and send them the appropriate advertisement packets. Finally, one could integrate the advertisement directly with the routing state propagation through the "Opaque LSA option" of OSPF link state advertisement messages, [6].

It is interesting to note that, even if a modification may be required in the routing daemon, the mechanism used to disseminate advertisement messages in the domain remain the sole decision of that domain's managers and that it will not affect how other domains disseminate their own advertisements, nor how a given client looks for a proxy.

We expect that the preferences for proxy selection will depend on the applications that actually use the proxy as well as on the actual function provided by the proxy agent. The requesting application, for instance, might not be interested in an exhaustive list of the agents found near the shortest path, but it could prefer to receive the closest agent without having to wait for a full round trip time. It might also want to restrict the final selection to agents holding a function-dependent parameter above/below a given threshold. Similarly, the proxy agents themselves could have different requirements with respect to the function that selects which advertisement is finally kept on a router, and which one is dropped.

In the proposed solution, [4], that flexibility is offered through a lightweight interpreter in routers. Advertisement and lookup packets contain a small bytecode program that expresses the behaviour we want to apply on the packet, such as early reporting, pre-filtering of the gathered information, advertisement selection policy, etc.
Before we can use WASP as a middleware for service discovery, however, it is required to introduce a simple level of authentication. We will assume that the equipments at the edge of the network are capable of flagging every packet that comes from the outside. Packets that aren't flagged are considered as "super-packets" that are granted read-write access to a part of the key space which is read-only for regular packets. This is mandatory if we want to prevent remote end-system to capture traffic using malicious forged service advertisements.

4 CONCLUSIONS

In this deliverable some proposals were presented for both routing and service discovery that would possibly support an ANA network environment. Dynamicity is an inherent characteristic of such a modern environment not only with respect to topological changes (e.g., network size increment, mobility of the nodes) but also due to the heterogeneity of the network and the autonomicity of its elements. In particular, information is expected to be delivered over different compartments (for the inter-compartment communication case) that may correspond to different network environments. Both routing and service discovery should be able to deal with such diverse environment in an efficient and scalable manner.

Routing and Service Discovery in self-organized networks are two complementary and critical functionalities. The dynamic nature of the network requires adaptation to the environment as well as specific robustness and trust mechanisms.

In this deliverable, we studied the issues and constraints related to routing and service discovery with respect to the ANA architecture.

As far as routing is concerned, we consider that a “one size fits all” solution is not feasible. We support the design of the routing functionally towards three complementary directions :

- Self-organization of basic routing components, in order to set-up the appropriate solution for a given context.
- Composition of routing protocols among compartments.

In this two above mentioned solutions, connectivity is enforced and a path is derived between the peer communicating devices. Service Discovery will use routing as a basic connectivity service.

- Content-based routing. In this last scenario, the Publish and Subscribe (PubSub) model will be supported and service discovery will be natively supported as, in the PubSub model, a service or content is advertised by a node and network support is provided to access the service.

Service discovery should be able to deal with the dynamicity associated with the ANA environment. This dynamicity prohibits centralized approaches (some of them presented earlier) since they do not scale (for example as the network size increases) and favors distributed, scalable and particularly adaptable approaches. As it was presented in this document there is a tight connection between the three service discovery phases that if exploited, may lead to smaller (overall) resource consumption as opposed to isolated treatment (and design) of each particular phase.

The complementary nature of the service discovery phases should be further studied and exploited for the efficient dealing with different network environments (e.g., when inter-compartment communication is considered). In ANA, the service discovery mechanism should be able to adapt to the idiosyncrasies of each particular network environment in order to provide for an overall scalable solution. For example, the number of service facilities in two different compartments may be different and therefore, the intensity of the advertising process may be different as well as the intensity of the searching process. Allowing the service discovery phases to adapt to different environments may be a suitable way to deal with the diverse environments.

Further performance optimization is also possible for service discovery if it is going to take into consideration the routing mechanism and especially information that may be available through the routing process. For example, a simple approach would be for routing messages to convey service information whenever they are injected in the network (and most likely when trying to identify a path towards a new node). This approach would enhance the advertising phase and therefore allow for higher probability of success for service discovery.

Another approach for locating services that can benefit from routing is transparently placing services in the network. In traditional network location theory, demand for service is assumed to originate from nodes of the network. Users that wish to obtain a service send requests to the servers in order to consume the requested service. Instead, services may be transparently sited in the network such as users do not need to be aware of their location. The location of services is kept unknown to the users that do not require to be explicitly configured with the location of the services. Demands for service originate not from the end-points but from flows traveling on various paths of the network. Servers will serve the flows that pass through them. The objective here will be not to optimize some objective functionally related to distance but to locate the services so as to maximize the total flow that are intercepted in the network. This approach contrasts with the three stages identified in service discovery since it considers the routing as an input to service placement rather than the opposite: the paths through which non-zero traffic flows have to be known.

Service discovery may also benefit routing in a similar manner by maintaining information about the path towards a particular destination that hosts a service. Imagine the worst case in which a searching for a service is initiated and its location becomes known after some message exchanges. Afterwards, if communication is to be established and there is no routing information about the path towards the particular node, the routing protocol should start a process to identify the path towards the particular node. This process will consume some extra network resources that may have been avoided, if the service discovery had only kept some information about the path towards the particular service.

5 REFERENCES

[1] D. Magoni and J. Pansiot. *Network layer search service using oriented multicasting*. In *21th IEEE INFOCOM*, pages 1346–1355, 2002. New York.

[2] D. Kosti et al. R. Braynard. *Opus: an overlay peer utility service*. In *5th IEEE OPENARCH*, pages 168–178, June 2002. New York.

[3] J. Touch and S. Hotz. *Dynamic internet overlay deployment and management using the x-bone*. In *ICNP*, pages 59–68, 2000. Osaka Japan.

[4] Sylvain Martin and Guy Leduc. *An active platform as middleware for services and communities discovery*. In *International Conference on Computational Science 2005*, number 3516 (part 3) in LNCS, pages 237–245, 2005.

[5] Sylvain Martin and Guy Leduc. *A dynamic neighbourhood discovery protocol for active overlay networks*. In *IWAN 2003*, number 2982 in LNCS, pages 151–162, 2003.

[6] R. Keller and B. Plattner. *Self-configuring active services for programmable networks*. In *IWAN 2003*, number 2982 in LNCS, pages 137–150, 2003.

[7] J.Eberspacher, R.Schollmeier, S.Zols, G.Kunzmann, *Structured P2P Networks in Mobile and Fixed Environments*.

[8] A.Iamnitchi and I.Foster, *A Peer-to-Peer Approach to Resource Location in Grid Environments*. In.J.W. wt.al., editor, Grid Resource Management. Kluwer Publishing 2003.

[9] V.Iyengar, S.Tilak, M.J.Lewis and N.B. Abu-Ghazaleh, *Non-Uniform Information Dissemination for Dynamic Grid Resource Discovery*.

[10] Christopher N. Ververidis, George C.Polyzos, *Extended ZRP: a Routing Layer Based Service Discovery Protocol for Mobile Ad Hoc Networks*, mobiquitous, pp. 65-72, The second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2005.

[11] Dipanjan Chakraborty, Anupam Joshi, Tim Finin, and Yelena Yesha. *GSD: A novel groupbased service discovery protocol for MANETS*. In 4th IEEE Conference on Mobile and Wireless Communications Networks (MWCN 2002.)

[12] S.Helal, N.Desai, V.Verma, C.Lee, *Konark. A service Discovery and Delivery Protocol for Ad-hoc Networks*, Proceedings of the Third IEEE Conference on Wireless Communication Networks (WCNC), New Orleans, March 2003.

[13] F. Sailhan, V.Issarny. *Scalable Service Discovery for MANET*. Proceedings of the IEEE International Conference on Pervasive Computing and Communications, Pages: 235-244 (2005).

[14] N. Laoutaris, V. Zissimopoulos, and I. Stavrakakis, *Joint Object Placement and Node Dimensioning for Internet Content Distribution*, Information Processing Letters, Vol. 89, No. 6, pp. 273-279, March 2004.

[15] http://en.wikipedia.org/wiki/Universal_Description_Discovery_and_Integration

[16] http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci508228,00.html

[17] http://www.openslp.org/doc/html/IntroductionToSLP/index.html

[18] http://en.wikipedia.org/wiki/Service_Location_Protocol

[19] http://en.wikipedia.org/wiki/Distributed_hash_table

[20] http://www.etse.urv.es/~cpairot/dhts.html

[21] http://www.usenix.org/events/osdi2000/full_papers/gribble/gribble_html/node4.html

[22] C. N. Ververidis, G. C. Polyzos, *Extended ZRP:Performance Evaluation of a Routing Layer Based Service Discovery Protocol for Mobile Ad Hoc Networks*, in the Proceedings of the Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'05), pp. 114-123, July 2005, San Diego, California.

[23] P.B. Mirchandani and R.L. Francis, "Discrete Location Theory," John Wiley and Sons, 1990.

[24] O. Kariv, and S.L. Hakimi, "An algorithmic approach to network location problems, II: The p-medians," SIAM Journal on Applied Mathematics, 37, 3 (1979), 539-560.

[25] A. Tamir, "An O(pn2) algorithm for p-median and related problems on tree graphs," Operations Research Letters, 19 (1996), 59–64.

[26] J.-H. Lin, and J.S. Vitter, "Approximation Algorithms for Geometric Median Problems," Information Processing Letters, 44:245-249, 1992.

[27] M. R. Korupolu, C. G. Plaxton, and R. Rajaraman, "Analysis of a local search heuristic for facility location problems," Proc. 9th Annual ACM-SIAM Symposium on Discrete Algorithms, 1-10, 1998.

[28] V. Arya, N. Garg, R. Khandekar, K. Munagala, and V. Pandit, "Local search heuristic for k-median and facility location problems," In Proceedings of the 33rd Annual Symposium on Theory of Computing (ACM STOC), pages 21–29. ACM Press, 2001.

[29] I.D. Baev and R. Rajaraman, "Approximation algorithms for data placement in arbitrary networks," In Proceedings of the 12th Annual Symposium on Discrete Algorithms (ACM-SIAM SODA), pages 661–670, January 2001.

[30] M. R. Korupolu, C. G. Plaxton, and R. Rajaraman, "Placement algorithms for hierarchical cooperative caching," In Proceedings of the 10th Annual Symposium on Discrete Algorithms (ACMSIAM SODA), pages 586-595, 1999.

[31] K. Oikonomou and I. Stavrakakis, "Scalable Service Migration: The Tree Topology Case," The Fifth Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net 2006), Lipari, Italy, June 14-17, 2006.

[32] L. Yamamoto and G. Leduc, "Autonomous reflectors over active networks: towards seamless group communication," AISB journal, special issue on agent technology, 1(1):125–146, December 2001.

[33] N. Laoutaris, G. Smaragdakis, K. Oikonomou, I. Stavrakakis, A. Bestavros, "Distributed Placement of Service Facilities in Large-Scale Networks", IEEE INFOCOM'07, May 6-12, 2007, Anchorage, Alaska.

[34] P. Erd¨os and A. R´enyi, "On random graphs I," Publ. Math. Debrecen, vol. 6, pp. 290–297, 1959.

[35] A.-L. Bar´abasi and R. Albert, "Emergence of scaling in random networks," Science, vol. 286, no. 5439, pp. 509–512, 1999.

[36] L. Subramanian, S. Agarwal, J. Rexford, and R. H. Katz, "Characterizing the internet hierarchy from multiple vantage points," in Proc. of IEEE Infocom '02, New York City, NY, USA, 2002.

[37] C. Gkantsidis, T. Karagiannis, P. Rodriguez, and M. Vojnovic, "Planet scale software updates," in Proc. of ACM SIGCOMM '06, Pisa, Italy, 2006.

[38] J. Pan, Y. T. Hou, and B. Li, "An overview dns-based server selection in content distribution networks," Computer Networks, vol. 43, no. 6, 2003.

[39] V. Lenders, M. May, and B. Plattner, "Density-based vs. proximity based anycast routing for mobile networks," in Proc. of IEEE Infocom'06, Barcelona, Spain, 2006.

[40] L. Tzevelekas and I. Stavrakakis, "Directed Budget-Based Clustering for Wireless Sensor Networks", 2nd International Workshop on Localized Communication and Topology Protocols for Ad-hoc Networks (LOCAN2006) held in conjunction with IEEE MASS 2006, October 9-12, 2006, Vancouver, Canada.

[41] R. Krishnan and D. Starobinski, "Efficient clustering algorithms for selforganizing wireless sensor networks," Ad Hoc Networks, vol. 4, no. 1, pp. 36–59, January 2006.

[42] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," in Proc. of ACM MobiComm'99, August 1999, pp. 263-270.

[43] H. Chan and A. Perrig, "Ace: An emergent algorithm for highly uniform cluster formation," in Proc. of IEEE European Workshop on Wireless Sensor Networks (EWSN'04). Springer Verlag, January 2004, pp. 154-171.

[44] L. Orecchia, A. Panconesi, C. Petrioli, and A. Vitaletti, "Localized techniques for broadcasting in wireless sensor networks," in Proc. of ACM DIALM-POMC'04, Philadelphia, USA, October 2004.

[45] Y. C. Tseng, S. Y. Ni, and E. Y. Shih, "Adaptive approaches to relieving broadcast storms in a wireless multihop mobile ad hoc network," IEEE Trans. Comput., vol. 52, no. 5, May 2003.

[46] Sylvain Martin and Guy Leduc. An active platform as middleware for services and communities discovery. In *International Conference on Computational Science 2005*, number 3516 (part 3) in LNCS, pages 237–245, 2005.

[47] Gkantsidis, C.; Mihail, M.; Saberi, A; “Random walks in peer-to-peer networks”, in proc. of INFOCOM 2004, Volume 1, 7-11 March 2004.

[48] Gkantsidis, C.; Mihail, M.; Saberi, A.; “Hybrid search schemes for unstructured peer-to-peer networks”, in proc. of INFOCOM 2005, Volume 3, 13-17 March 2005.

[49] Sanjay Shakkottai; “Asymptotics of query strategies over a sensor network”, in proc. of INFOCOM 2004, Volume 1, 7-11 March 2004.

[50] Ziv Bar-Yossef, Roy Friedman, Gabriel Kliot, “Location and membership services: RaWMS -: random walk based lightweight membership service for wireless ad hoc network”, Proceedings of MobiHoc '06, May 2006.

[51] Clip2 Distributed Search Solutions. « The gnutella protocol specification v0.4 », 2001.

[52] A. Rowstron, P. Druschel, “Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems”, IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany, pages 329-350, November, 2001

[53] T. Clausen and P. Jacquet, “Optimized Link State Routing Protocol (OLSR)”, rfc 3626, http://www.ietf.org/rfc/rfc3626.txt

[54] C. Perkins, E. Belding-Royer, S. Das, “Ad hoc On-Demand Distance Vector (AODV) Routing”, RFC 3561, http://www.ietf.org/rfc/rfc3561.txt

[55] Z. J. Haas, M. R. Pearlman, P. Samar, “The Zone Routing Protocol (ZRP) for Ad Hoc Networks”, Internet Draft, http://www.ietf.org/proceedings/02nov/I-D/draft-ietf-manet-zone-zrp-04.txt

[56] J. Moy, RFC-1131 The OSPF Specification, October 1989.

[57] J. Moy, RFC-2328 OSPF Version 2, August 1998.

[58] C. Hedrick, RFC-1058 Routing Information Protocol, June 1988.

[59] G. Malkin, RFC-2453 RIP Version 2, November 1998.

[60] K. Lougheed, Y. Rekhter, RFC-1105 A Border Gateway Protocol (BGP), June 1989.

[61] Y. Rekhter, T. Li, RFC-1771 A Border Gateway Protocol 4 (BGP-4), March 1995.

[62] Intermediate System to Intermediate System Intra-Domain Routing Exchange Protocol”, ANSI X3S3.3/87-150R, October 1987.

[63] C. Perkins, RFC-3344 IP Mobility Support for IPv4, August 2002.

[64] E. Nordmark, M. Bagnulo, Internet-Draft Level 3 multihoming shim protocol - draft-ietf-shim6-proto-07.txt, November 2006.

[65] R. Moskowitz, P. Nikander, RFC-4423 Host Identity Protocol (HIP) Architecture, May 2006.

[66] Charles E. Perkins und Pravin Bhagwat, “Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers”, ACM SIGCOMM’94

[67] David B. Johnson, David A. Maltz, Yih-Chun Hu, “The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)”,

[68] D. Estrin, R. Govindan, J. Heidemann and S. Kumar, “Next Century Challenges: Scalable Coordination in Sensor Networks “, In Proceedings of the Fifth Annual International Conference on Mobile Computing and Networks (MobiCOM ‘99), August 1999, Seattle, Washington (45)

[69] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton and J. Zhao, “Habitat monitoring: Application driver for wireless communications technology”, ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean, Costa Rica, April 2001

[70] S. Tilak, N. Abu-Ghazaleh, W. Heinzelman, “A Taxonomy of Wireless Micro-Sensor Network Models”, ACM Mobile Computing and Communications Review, April 2002

[71] A. Boulis, S. Ganeriwal, and M. Srivastava. Aggregation in sensor networks: An energy-accuracy trade-off. In Proc. IEEE SNPA, Anchorage, AK, May 2003.

[72] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, “Wireless sensor networks: a survey”, Elsevier Computer Networks, 2002.

[73] C. Intanagonwiwat, R. Govindan, D. Estrin, “Directed Diffusion: A scalable and Robust Communication Pradigm for Sensor Networks”, In Proceedings of the Sixth Annual International Conference on Mobile Computing and Networks (MobiCOM 2000), August 2000, Boston, Massachusetts (46).

[74] S. Hedetniemi and A. Liestman, “A survey of gossiping and broadcasting in communication networks”, vol. 18, no. 4, pp. 319-349, 1988.

[75] W. Heinzelman, A. Chandrakasan, H. Balakrishnan, “Energy-efficient Communication Protocol for Wireless Microsensor Networks”, Proc. of the Hawaii International Conference on System Sciences, Jan 2000.

[76] K. Akkaya and M . Younis, “A Survey on Routing Protocols for Wireless Sensor Networks”, in Elsevier Ad Hoc Networks Journal, vol. 3/3, pp. 325-349, 2005.

[77] V. Rodoplu and T.H. Ming, “Minimum energy mobile wireless networks”, IEEE JSAC, vol. 17, no. 8, pp. 1333-1344, 1999.

[78] W. Rabiner Heinzelman, J. Kulik, and H. Balakrishnan, “Adaptive Protocols for Information Dissemination in Wireless Sensor Networks,’’ Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom ‘99), Seattle, Washington, August 15-20, 1999, pp. 174-185.

[79] Kevin Fall. A Delay Tolerant Networking Architecture for Challenged Internets Proc. SIGCOMM 2003, Aug. 2003

[80] Amin Vahdat and David Becker. Epidemic routing for partially connected ad hoc networks Technical Report CS-200006, Duke University, April 2000.

[81] J. Shoch , “Inter-network naming, addressing, and routing”, in Proceedings of IEEE Computer Conference, COMPCON (Washington, D.C., Fall). IEEE, New York, 1978.

[82] D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja1, J. Pruyne, B. Richard, S. Rollins, Z. Xu, “Peer-to-Peer Computing”, report, HP Laboratories Palo Alto, July 2003.

[83] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek et H. Balakrishnan, “Chord: A scalable peer-to-peer lookup service for internet applications,” IEEE/ACM Transactions on Networking , vol. 11, no. 1, pp. 17–32, february 2003.

[84] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, “A Scalable Content-Addressable Network”, In Proceedings of ACM SIGCOMM 2001.

[85] B. Y. Zhao, J. D. Kubiatowicz and A. D. Joseph, “Tapestry: An infrastructure for fault-tolerant widearea location and routing,” tech. rep., University of California, Berkeley - UCB/CSD-01-1142, EECS, May 2001.

[86] R. Baldoni, R. Beraldi, L. Querzoni, G. Cugola and M. Migliavacca “Content-Based Routing in Highly Dynamic Mobile Ad Hoc Networks”, in the proceedings of the IEEE International Conference on Pervasive Services 2005 (ICPS’05),, Santorini, Greece, July 2005.

[87] R. Baldoni, R. Beraldi, G. Cugola and M. Migliavacca, “Structure-less Content-Based Routing in Mobile Ad Hoc Networks”,

[88] E. Yoneki and J. Bacon. An adaptive approach to content-based subscription in mobile adhoc networks. InProceedingsof theSecondIEEEAnnual Conference on Pervasive Computing and Communications Workshops (PERCOMW04).IEEE,2004.

[89] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, “A scalable content-addressable network”. In Proc. ACM SIGCOMM 2001, August 2001.

[90] Carzaniga, A.; Rutherford, M. J. & Wolf, A. L, “A Routing Scheme for Content-Based Networking”, Proceedings of IEEE INFOCOM 2004, volume 2, pages 918-928.

[91] Cristophe Jelger, Stephan Schmid and al, “ANA Blueprint-First Version”, Technical Report for the ANA project, January 2007, “www.ana-project.org”

[92] James P.G. Sterbenz, Rajesh Krishnan, Regina Rosales Hain, Alden W. Jackson, David Levin, Ram Ramanathan, and John Zao, “Survivable Mobile Wireless Networks: Issues, Challenges, and Research Directions”, *Proceedings of the ACM Wireless Security Workshop (WiSE) 2002 at MobiCom*, Atlanta GA, September 2002, pp. 31–40

APPENDIX A: A FEW PRELIMINARY RESULTS

Part I

Suppose that the network topology is represented by an undirected graph $\mathrm{G(V, E)}$, where V is the set of nodes and E the set of links among them. Let S_v denote the set of nodes that have a direct link with node v. Let the edges of the graph be assigned a positive integer referred to as *weight*. Let $d_{u,v}$ denote the *distance b*etween node u and node v, corresponding to the summation of the weights along a *shortest path* among the two nodes (for the same node $d_{v,v} = 0$). Alternatively, $d_{u,v}$ denotes the *traveling cost* between node u and node v. Let λ_v denote the rate at which data packets are transferred through the network between node v and the service node for the particular service: λ_v will be referred to as the *service demands* of node v. Let X_k be the set of k nodes at which the service is located. That is, it is assumed that there are k nodes which are capable of providing a given service. For a given placement X_k of these nodes and assuming that the cost of service provision is directly proportional to the amount of data transferred per unit time (λ_v) and the distance traveled ($d_{u,v}$), the total cost of service provision, $C(X_k)$, is given by, $X_k \in Y_k$,

$$C(X_k) = \sum_{\forall v \in V} \lambda_v \min_{u \in X_p} \{d_{v,u} : u \in X_k\},$$

where, Y_k is the set of all possible k*-placements*, [23]. The solution of the k*-median problem* amounts to determining the placement X_k such that $C(X_k)$ is minimized. Figure 3 depicts a graphical representation of a 2-median problem (the service is located at nodes y and δ denoted by the dotted ellipses).

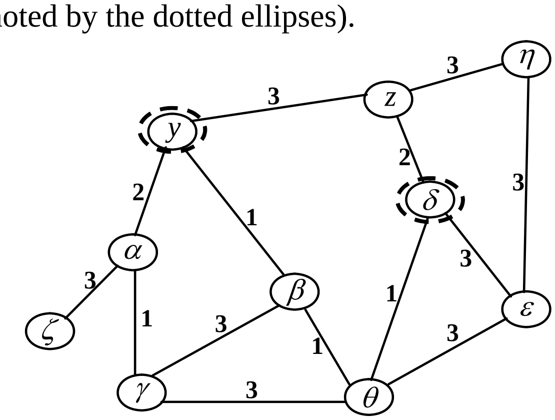

Figure 3: Example of a 2-median problem for a certain network topology.

Part II

Simulator Description

The simulator is a proprietary program written in C. The considered network is a random graph (Erdos-Renyi, [34]), and has been created for various values of the probability p_{ER}. Note that for high values of p_{ER}, the resulting random graph is very close to the fully connected graph. For $p_{ER} \geq 1/N$, the graph is connected with high probability and vice versa.

The advertising process will take part initially at the service node. This node will start the advertising process, by passing the information, about the location of the service, to some of its one hop neighbors, in the first timeslot, and so on, according to probability p_{fl} that has been defined at the beginning of the simulation. The flooding will continue by the nodes that learn about the location of the service. These nodes will pass the information (according to the same probability p_{fl}) to their neighbors, except from the one that gave it to them. The flooding will continue until the moment where a node doesn't have a message to send to its neighbors either because it has no neighbor to inform or because of being prohibited by the p_{fl} probability.

At the end of the advertising process there will be a number of nodes who know about the location of the service and others that are completely unaware of it. From the above process, the results derived concern the number of messages it takes to complete the advertising, the percentage of the nodes that have information about the location of the service, as well as the number of timeslots it takes to end the process are examined with regard to probability p_{fl}.

Simulation Results

At this point we present some of the derived simulation results. For most of the following figures the number of nodes in the network is $N = 1000$. The number of messages exchanged until the end of the advertising process is an interesting metric for the overall performance as well as the percentage of nodes that have been informed about the location of a particular service.

In Figure 4, the total number of messages for a network of 1000 nodes is depicted as a function of p_{fl}. Note that for any network size there exist a wide range of corresponding random graphs depending on the particular value of p_{ER}. As it is clearly depicted in Figure 4, for small values of p_{ER}, the number of exchanged messages is rather small. This is expected since for small values of p_{ER}, the network is not connected with high probability (it looks like clusters of isolated nodes). Consequently, the advertising process that is initiated by the node at which the service is located, will not have the chance to reach more nodes of the network than the limited number that consists of its cluster, even for large values of p_{fl}.

For values of p_{ER} greater than $1/N$, it is possible to observe an almost linear increment of the total number of messages as p_{fl} increases. It is interesting to observe the behavior of the curves for p_{ER} close to $1/N$. It is observable that the total number of messages remains small (almost zero) until a certain value of p_{fl} at which the total number of messages rapidly increases and afterwards it follows an almost linear increament.

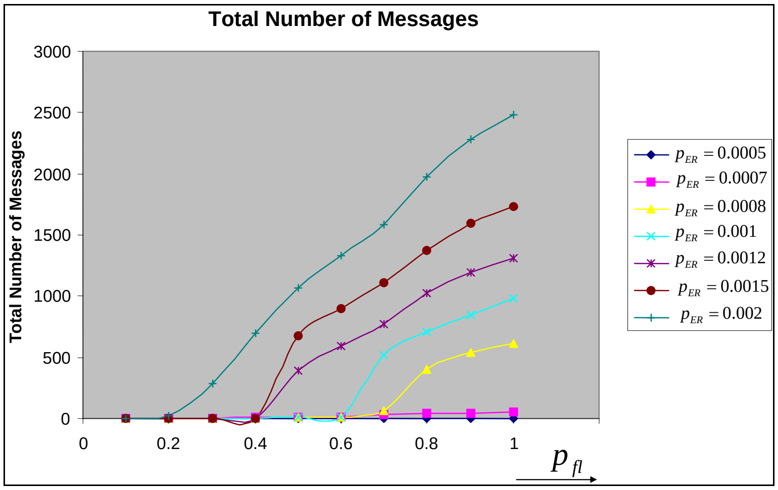

Figure 4: Total number of messages as a function of p_{fl} for different random graphs (different values of p_{ER}) for a network of 1000 nodes.

This rapid increase when p_{ER} is set close to $1/N$, can be explained if we consider the fact that for $p_{ER}N = 1$ the *phase transition* phenomenon takes place in random graphs and the entire network suddenly acquires some properties (e.g., connectivity) that were not present before (with high probability).

In Figure 5, the percentage of informed nodes is depicted as a function of p_{fl} for the same network as before. As it would be expected, the higher the number of messages sent, the higher the percentage of informed nodes.

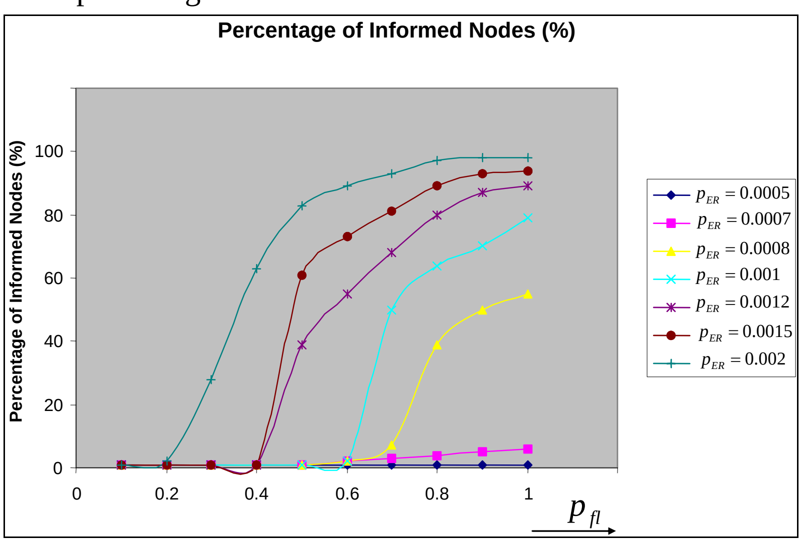

Figure 5: Percentage of informed nodes as a function of p_{fl} for different random graphs (different values of p_{ER}) for a network of 1000 nodes.

There are a number of important results that could be extracted form Figure 5 considering Figure 4. For example it is possible to observe that the same degree of the percentage of informed nodes may be achieved for smaller number of messages for the same network. Take for example the network that has been created for $p_{ER} = 0.002$. From Figure 5 it is evident that for p_{fl} from 0.8 to 1.0, the percentage of the informed nodes is around 100%. For p_{fl} equal to 0.5, the percentage of the informed nodes is close to 80%, which is rather satisfactory (note that the scope of advertising is not to inform the entire network but rather a small suitable proportion). For p_{fl} equal to 0.6 the percentage of informed nodes is close to 85%.

From Figure 4 for the same network ($p_{ER} = 0.002$), it is possible to see that for p_{fl} equal to 1.0, 0.9 and 0.8, the number of messages is close to 2500, 2300 and 2000 respectively. Consequently, almost 100% of the nodes can be informed when p_{fl} is set to 0.8 and "save" almost 25% of messages, compared to the case of p_{fl} equal to 1.0. Furthermore, for p_{fl} equal to 0.5 (percentage of informed nodes close to 80%), the number of messages required is close to 1000, which is a significant saving compared to 25000 when p_{fl} is set to 1.0. Note that 80% is still a large proportion of the overall network (it means that 800 out of the 1000 nodes have been informed). When the requirement is that 30% percent of the nodes of the network are informed, then from Figure 5, this corresponds to p_{fl} close to 0.3, and the number of messages required to achieve this (according to Figure 4) is almost 350.

Note that 30% of informed nodes correspond to 300 nodes in a network of 1000 nodes. Since 350 messages were required to inform these nodes, it is apparent that almost all messages (300/350*100=85%) were used to inform new nodes while the rest of were informing nodes were informed by other. This is rather efficient compared to the case for p_{fl} equal to 1.0, where 1000/2500*100=40% of the messages informed new nodes.

The same observations can be drawn for any other network (different values of p_{ER} and N). Depending on the particular case, the selection of the appropriate value of p_{fl} is rather important for the achievement of a certain requirement. For example, when p_{ER} is set to 0.002, 30% of the nodes can be informed for p_{ER} equal to 0.3. However, this is not the case when p_{ER} is set to 0.001.

Consequently, the appropriate value for p_{fl} depends of the particular topology characteristics (for the case of random graphs on N and p_{ER}). Therefore, under ANA we investigate this particular area attempting to derive further simulation results but also analytical results that would allow for the investigation of a wide range of networks.

Note that the proposed probabilistic flooding is not the only approach that will be considered in ANA. Other approaches (like random walks, agent leashing etc.) may also be considered.

Part III

Simulation Results

There are many figures containing simulation results depending of the range of possible scenarios. However, the most important result for the search for the service is the success percentage for the aforementioned network of 1000 nodes.

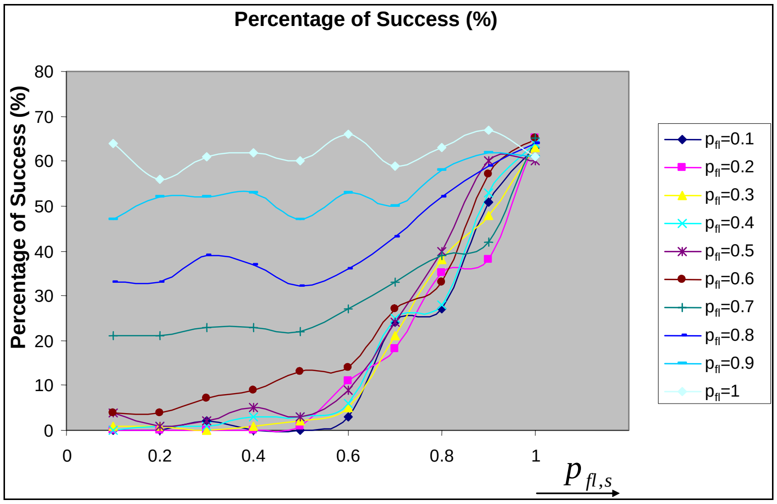

Figure 6: Search for the service percentage of success as a function of $p_{fl,s}$ for different values of p_{fl} (different advertising process) for a random graph of 1000 nodes, where $p_{ER} = 1/N$.

It is possible to observe from Figure 6 that the maximum possible corresponds to around 60%. This particular upper bound is in compliance with the percentage of informed nodes during the advertising process (for $p_{ER} = 1/N$, when N=1000). If p_{fl} is equal to 1.0, it is easy to see that irrespectively of $p_{fl,s}$, this upper bound is always achieved. When p_{fl} is smaller than 1.0, but not that small (higher than 0.8), the percentage of success, even though not that high is comparably close to the upper bound, even for rather small values of $p_{fl,s}$. As $p_{fl,s}$ increases then percentage of success increases as well.

For small values of p_{fl}, however, the success percentage is rather small for values of $p_{fl,s}$ smaller than 0.6. As $p_{fl,s}$ increases, then the success percentage increases and it achieves the upper bound when $p_{fl,s}$ is close to 1.0.

There are some interesting observations that could be drawn from this particular figure in conjunction with the previous figures. It has been demonstrated that the upper bound for the percentage of success is dominated by the outcome of the advertising process and the particulars of the network topology (for the demonstrated case there was a “huge” connected cluster consisting of 60% of the network nodes). It is also shown that the upper bound is achievable when the searching process is ready to search the entire network. However, if a smaller than the upper bound percentage of success is acceptable, a smaller number of messages will be needed.