Objective FP7-ICT-2007-1-216041/D-4.5

.

The Network of the Future

Project 216041

"4WARD – Architecture and Design for the Future Internet"

D-4.5
Evaluation of the in-network management approach

Date of preparation: **10-06-11**
Start date of Project: **08-01-01**
Project Coordinator: **Henrik Abramowicz**
Ericsson AB

Revision: **1.0**
Duration: **10-06-30**

Document Properties[1]:

Document Number[2]:	FP7-ICT-2007-1-216041-4WARD / D-4.5
Document Title:	**Evaluation of the in-network management approach**
Document responsible:	Fabian Wolff, FhG.
Author(s)/editor(s):	Andrei Bogdan Rus, Virgil Dobrota, Emanuel Puschita, Tudor Palade, (TUCN), Christopher Foley (TSSG), Rebecca Steinert, Daniel Gillblad (SICS), Alberto Gonzalez, Rolf Stadler. Mads Dam, Fetahi Wuhib, Karl Palmskog (KTH), Susana Sargento, Lucas Guardalben (IT), Vitor Mirones (PTIN), Giorgio Nunzi, Dominique Dudkowski, Marcus Brunner (NEC), Thomas Hirsch, Jens Tiemann, Cristián Varas, Fabian Wolff (Fraunhofer), Catalin Meirosu (EAB), Slawomir Kuklinski (TPSA), Changpeng Fan, Henning Sanneck (NSND), Leonard Pitu (SROM)
Target Dissemination Level[3]:	PU
Status of the Document:	Final
Version	1.0

Revision History:

Revision	Date	Issued by	Description
1.0	2010-06-11	Fabian Wolff	First public version

This document has been produced in the context of the 4WARD Project. The research leading to these results has received funding from the European Community's Seventh Framework Programme ([FP7/2007-2013] [FP7/2007-2011]) under grant agreement n° 216041

[1] Input of Title, Date,Version, Target disseminationlevel, Status via *"File /Properties/Custom*" in the Word menu
[2] Format: FP7-ICT-2007-1-216041-4WARD /<Deliverable number>
Example: FP7-ICT-2007-1-216041-4WARD / D-1.1
[3] Dissemination level as defined in the EU Contract:
PU = Public
PP = Distribution limited to other programme participants
RE = Distribution to a group specified by the consortium
CO = Confidential, only allowed for members of the consortium

Abstract

In-Network Management, INM, is a novel network management concept designed by WP4, in which management tasks are embedded in the network, while utilizing distributed architecture, self-organization and autonomy. This deliverable evaluates the INM design work recently completed by all WP4 tasks and cross WP activities. It compiles a comprehensive list of requirements collected from the definition of Future Internet scenarios that were described at the beginning of the project, and use it as an evaluation criterion. An adapted V-model is used for the evaluation methodology, with two agreed templates, one for the framework and one for the algorithms.

The INM evaluation analysis demonstrates a comprehensive coverage of requirements. INM is shown to be beneficial for all evaluation criteria, NewAPC, VNets, GPs, and NetInf, while realising potential business incentives, when compared with legacy network management systems. As the next step, INM should be experimented with real networks.

Keywords

Future Internet, in-network management, self-management, real-time management, scalable and robust management systems, architectural elements, situation awareness, self-adaptation, prototype, evaluation, v-model

Table of Contents

1 Executive summary

Work Package 4 (WP4) works towards the definition of novel management instruments to operate the future Internet. In-Network Management, (INM), utilizes decentralization, self-organization and autonomy as its basic enabling concepts. The idea is that, contrary to the legacy centralized approach, the management tasks are embedded in the network. The managed system now executes management functions on its own. The INM concepts and its design are detailed in 4WARD WP4 deliverables [23] [24].

Deliverable D4.1 [22] described scenarios and use cases for the Future Internet, and for each scenario, derived requirements to enable it. This deliverable compiles a comprehensive list of requirements from all the scenarios, and uses them as a basis for evaluating the INM design concepts developed by all WP4 tasks.

The evaluation process follows a WP4-adapted V-model, in which the INM implementation is checked against its testing results in a top-down approach, from the full system down to each of its components. This methodology facilitates an evaluation without the need for a comprehensive implementation of all NM functionality, a valuable feature for this clean-slate conceptual project. The evaluation effort is split into three separated topics that match the structure of the WP4 activities: framework, algorithms and a demonstrator. Depending on the extent of the implementation, different evaluation instruments are used.

The analysis utilizes two agreed templates, one for the framework and one for the algorithms. The framework is evaluated for scalability, robustness, reduced integration effort, and reduction of complexity. Each INM algorithm from tasks 4.3 and 4.4 is evaluated with the algorithm template.

Special attention was given for the evaluation of cross WP activities: INM for NewAPC (cross WP2/4), INM for VNET (cross WP3/4), INM for GPs (cross WP4/5), and INM for NetInf (cross WP4/6). The business aspects of INM (includes cross WP1/4) were also studied.

The evaluation analysis demonstrates that every requirement identified in [22] was addressed by some algorithms in [23] and [24]. The degree of coverage of each requirement varies, and explanations are given for those that are lightly addressed. Compared with legacy management systems, INM design is shown to be scalable and robust. Moreover it facilitates reduction of integration effort and reduction of complexity. Most importantly, INM is beneficial for NewAPC, VNets, GPs and NetInf, and it demonstrates business incentives that are realized with potentially reduced OPEX and increased EBITDA values.

In Summary, the evaluation analysis of the simulation results shows that the INM design is beneficial for the Future Internet. The next step is to test the feasibility of INM concepts in real experimental networks.

2 Terminology

Anomaly detection	Analysis of network end-to-end measurements, deviating from normally observed behaviour.
Atomic INM algorithm	An INM algorithm that cannot be decomposed into smaller parts without losing functionality. Faults and anomalies are detected in a distributed manner, involving collaborative fault-localisation.
CAPEX	Capital Expenditures are expenditures creating future benefits.
CLQ	CLQ (Cross-Layer QoS) is a management capability that runs permanently to monitor the infrastructure performances on top of the MAC Sub-Layer (One-Way Delay, Available Transfer Rate). It accepts also service requests (i.e. committed QoS parameters, network status, etc.)
Collaborative fault-localisation	Isolation of abnormal behaviour to certain network components.
Co-Design	Style of designing management functions in conjunction with service functions.
(M)DHT	Distributed Hash Table
Drive Testing	Analysis of wireless mobile access quality (coverage, capacity) by using a vehicle with test equipment
DSL	Domain Specific Language
EBITDA	Earnings before interest, taxes, depreciation, and amortization.
FCAPS	Fault and Configuration Management, Accounting Management & User Administration, Performance and Security Management.
Flooding	Simple routing or distribution algorithm in which every incoming packet is sent through every outgoing link
Generic Aggregation Protocol (GAP)	Distributed algorithm that provides continuous monitoring of global metrics and supports qualitative accuracy objectives.
Generic Path (GP)	Research topic of Workpackage 5 within 4WARD
Global Management Point (GMP)	High-level entry point via which a network is managed in terms of high-level objectives and according to the INM paradigm.
Global metric (or network-wide metric)	Result of computing a multivariate function, whose variables are local metrics from nodes across the networked system. Examples include the total number of VoIP flows in a domain and the list of the 50 subscribers with the longest end-to-end delay.

Gossiping	Gossiping for network exploration combines flooding and random walks in a tradeoff between messaging overhead and coverage.
I-NAME	In-Network Autonomic Management Environment is an algorithm that works in the self-organizing management plane as a resource management function and offers services to VNet, which generates virtual networks, by negotiating the QoS parameters inside the established virtual resources.
IMPEX	Implementation Expenditures; see also CAPEX and OPEX.
In-Network Management (INM)	Research topic of Workpackage 4 within 4WARD
LTE	Long Term Evolution, a standard within the 3GPP to improve the UMTS standard. Goals include improving efficiency, lowering costs, improving services, making use of new spectrum opportunities, and better integration with other open standards.
Management Capability (MC)	The building blocks for composing any basic and any more complex management functions from management algorithms.
Management Domain	Specific view on a set of self-managing entities, either structural or functional, providing access to a restricted set of management functions only.
NATO!	"Not All aT Once!", a statistical probability scheme and algorithms for precisely estimating the size of a group of nodes affected by the same event, without explicit notification from each node, thereby avoiding feedback implosion.
Network of Information (NetInf)	Research topic of Workpackage 6 within 4WARD
New Architectural Principles and Concepts (NewAPC)	Research topic of Workpackage 2 within 4WARD
P2P	Peer-to-peer overlay networks
Policy	A set of considerations that are designed to guide the decisions that affect the behaviour of a managed resource.
Quality of Service (QoS)	A set of quality requirements on the collective behaviour of one or more objects (ITU). In the field of networking term refers also to resource reservation control mechanisms rather than the achieved service quality.
Radio Access Network (RAN)	A RAN is a network for wireless access comprising of transceivers and base station controllers connected in a radio network infrastructure excluding the core network
Random Walk	Random walks describe a trajectory for exploration of a dynamic networks based on random steps

Remote Procedure Call (RPC)	A style of inter-process communication that can be used to implement distributed INM functions, e.g. on the level of management capabilities.
Representational State Transfer (REST)	A style of software architecture based on resources and a simplified set of functions to access such resources in a distributed way.
Self-Managing Entity (SE)	A component of a system that is self-managed by objective and can autonomously perform a series of management-related tasks, e.g. self-configuration and self-healing
Self-adaptation control loop	An algorithm or a portion of an algorithm within an MC that implement self-adaptation functionality for an INM algorithm.
Self-Managing Entity (SE)	A component of a system that is self-managed by objective and can autonomously perform a series of management-related tasks, e.g. self-configuration and self-healing.
self-x / self-*	A short form of the self-management paradigm, where 'x' represents the different occurrences, e.g. self-planning, self-configuration, self-optimization, self-tuning, self-self-healing, etc
Self-Organising Network (SON)	A concept of 3GPP to improve the self-management of LTE networks, including self-configuration, -optimization and -healing
Threshold Gossip-Generic Aggregation Protocol (TG-GAP)	Gossip-based protocol that detects the crossing of a configurable threshold by a global metric.
Virtual Network (VNet)	Research topic of Workpackage 3 within 4WARD

3 Introduction

New management instruments for the future Internet have been proposed within the scope of Work Page 4 (WP4) with a main focus on a framework and a set of distributed management algorithms. Due to limitations of today's centralized network management operations described in our previous deliverable D4.1 [22], WP4's work follows a new decentralized network management approach, called In-Network Management (INM). A detailed technical description of this approach has been presented in the deliverable D4.3 [24].

3.1 Objective of document

The objective of this deliverable is to evaluate the work done by WP4 within the last 30 months. To be more precise, the developed solutions will be evaluated against the requirements for an INM approach, derived from the scenarios described in D4.1.

The following topics will be evaluated regarding their suitability to enable INM for the Future Internet.

- **INM framework**
- **Distributed management algorithms**
- **Collaboration with other WPs** within 4WARD
- **Business values**

3.2 Structure of document

Chapter 4 gives an overview about the used approaches and tools of the evaluation. It also introduces a template used for the evaluation of the INM framework and algorithms. Chapter 5 evaluates the INM framework with respect to the requirements scalability, robustness, integration effort, and complexity. Chapter 6 evaluates how specific distributed management algorithms cover functional requirements to INM. Chapter 7 describes an evaluation of INM in collaboration with other WPs. Chapter 8 presents an evaluation of the business values that INM provides. Finally, chapter 9 concludes the deliverable.

4 Overview on Evaluation

This chapter gives a short overview on the evaluation process and requirements coming from previous work. This deliverable picks up and extends work on scenarios and use cases, documented in 4WARD Deliverable D4.1 [22]. Based on evaluation templates originating from a common approach, the framework and distributed management algorithms are evaluated based on requirements derived from D4.1.

4.1 Requirements according to D4.1

4WARD Deliverable D4.1 [22] describes 4 scenarios and several use cases which lead to a number of requirements that need to be fulfilled by the INM architecture and distributed management algorithms. These are:

- Scenario 1: Self-Management in wireless multi-hop networks
- Scenario 2: Large Operator
- Scenario 3: Home Networks
- Scenario 4: DEFCON (Large scale adaptation in response to dramatic events)

Each of the scenarios leads to specific use cases and requirements for the INM. Some of the requirements are addressed in more than one scenario, so a structuring of the requirements makes sense. Following the information gathering and processing for network management purposes, the following requirements were collected as a guideline for evaluation of developed INM components (the original scenarios are shown in brackets):

Information Gathering and Collection

- Monitoring of lower layer info (SNR, link state …) (1)
- Situation awareness, detection of network conditions (1, 4)
- Detection of network anomalies (2)
- Characteristics of devices (3, 4)
- Information about network resources (2)

Information Distribution and Node Collaboration (by Node Interaction)

- Common Information model and protocols (2, 4)
- Distribution of captured and collected information (1)
- Multicasting status and capabilities of nodes (3)
- Information exchange done in a standardized way (1)
- Distributed management and role based interaction (2)
- Security Framework, establishment of trust relationships (3, 4)

Core Network Management Requirements

- Management based on situation, policies and/or business objectives (2, 4)
- Self-Management e.g. according to FCAPS model (2, 3)
- Self-Adaptation of network components (4)
- Distributed Network Management architecture (3, 4)
- Make decisions and take actions (2)

Special, Data Communication Related Requirements

- Traffic differentiation and handling (2, 3)
- Routing related functionality: Route discovery (2) or self-rerouting (3)

Performance of Network Management Mechanisms

- Reduction of computation load on each node by using distributed management (2)
- Reduction of management information flow by using distributed management (2)
- Reliability at least similar to centralized network management approach (2)
- Substantially faster adaptation compared to centralized approach (2)
- Quick switching of network wide behaviour (4)
- Self-adaptive behaviour optimizes available resources (1)

4.2 Evaluation Process

The evaluation of parts of a system or a system as a whole is an integral part of the development process. Many different software development and, evaluation methods are described in literature and used in real project work, like the spiral or the waterfall model. To illustrate and motivate the approach applied here (and described in the 4WARD project description) the V-model [31] is used. The V-model illustrates on x-axis the time and on y-axis the detailing of the implementation. The left side represents the implementation while the right side represents testing of the system and its components. In other words: the implementation of a system follows a top-down approach, starting with the system architecture, defining function blocks and implementing these functions. Single functions are tested separately and afterwards the system is integrated from these functions, while tests cover more and more parts of the final system. By relating the left side of different steps of system implementation with needed tests on right side it is possible to bring the analysis of the final system forward. Interfaces and interaction can be tested (possibly in a different environment and/or with special tools) without implementing the full functionality, which allows an early adaption to design issues and a less expensive redesign of system parts.

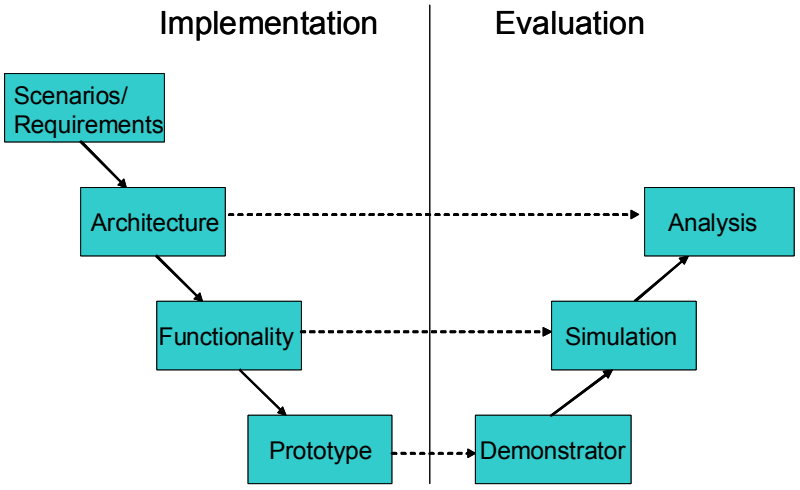


Figure 4-1: System Design & Implementation and Relation to Evaluation

However, as there will not be a complete implementation and there are several, partly independent parts of the INM developed within 4WARD WP4 the V-model cannot be fully

applied in its original form, e.g. there is no need of an operation and maintenance step. As shown in Figure 4-1 we adapted the V-model in such a way that it matches the structure and needs of WP4's work, namely logically splitting the work items into the three topics framework, algorithms and demonstrator. To summarize this approach Figure 4-2 visualizes, which evaluation instruments (as mentioned in the 4WARD project description) are used in the evaluation process at a specific step.

Evaluation Instrument	Analysis	Simulation	Demonstrator
Evaluation Metrics & Criteria	Requirements to the INM Architecture ⇩ General Feasibility	Key Performance Indicator (KPI), e.g. throughput, delay ⇩ Increased Performance	Based on objectives ⇩ Proof-of-Concept
Evaluated topics	INM Architecture, Situation Awareness	Algorithms for Adaptation & Monitoring	Demonstrator

Figure 4-2: Evaluation Instruments

The table in Figure 4-2 shows how the depth of implementation is reflected by different evaluation instruments: The high-level and abstract parts of INM features are evaluated by analysis, which describes the interaction of components, interfaces and functions to fulfil certain requirements. The result is a general feasibility on conceptual and architectural level. Additional evaluation steps are needed to get more concrete results. Simulations of single, separate components and functions show gains like improved performance by applying the new INM principles. The demonstrator plays the role of the final system (compared to the usual development process), which integrates all functionality. In this research project we show a proof of concept based on selected functionality. In this deliverable the Demonstrator is covered only by a short overview, the full description of the demonstrator can be found in deliverable D4.4 [25].

4.3 Evaluation Templates

As mentioned earlier the evaluation of the framework and the distributed management algorithms are done in a structured way. Inspired by use case templates, the evaluation template is structured into different sections. For the last step of evaluation, a lean version of a template (compared to use case templates) was developed, which still contains typical and necessary sections. The following subsections present the structure of the used evaluation template.

4.3.1 Evaluation template for Framework

The evaluation of the framework will follow the structure of Figure 4-3.

Name	Name of the Requirement
Description	Description of the Requirement
Assumptions	Preconditions and description of environment
Use Case	Evaluated Use Case
Actions	Transition during evaluation to see expected behaviour
Metric	Description of metric or KPI which captures the result
Evaluation	Based on analysis and simulations

Figure 4-3: Overview on Evaluation Template for the INM Framework

Name and *Description* give an overview of the problem space of the evaluated requirement. Section *Assumptions* describes general preconditions that must be fulfilled. Next sections *Use Case* and *Actions* specifically describe the expected network environment of the evaluated requirement. A description of *Metrics* is important if the requirement is evaluated in such a way that the result are quantitative, e.g. based on simulation results. Lastly, section Evaluation describes the result of the evaluated requirement, which can be done analytically or simulation based.

4.3.2 Evaluation template for Distributed Management Algorithms

The evaluation of the distributed management algorithms will follow the structure of Figure 4-4

Name	Name of the distributed management algorithm (according to naming of D4.3)
Description	Short description of this algorithm
Assumptions	Preconditions and description of environment
Requirement	Evaluated requirement according to list derived from D4.1
Actions	Transition during evaluation to see expected behaviour
Metric	Description of metric or KPI which captures the result
Evaluation	Based on analysis and simulations

Figure 4-4: Overview on Evaluation Template for Distributed Management Algorithms

Inspired by use case templates, the evaluation template is structured into different sections. For the last step of evaluation, a lean version of a template (compared to use case templates) was developed, which still contains typical and necessary sections. *Name* and *Description* section are for housekeeping and offer a short introduction into the topic. The section *Assumptions* gives an overview on preconditions for execution of this evaluation step, like a well-defined state of the system. Together with a description of the environment and possibly a trigger, this section summarizes all necessary information the evaluation is based on. The section *Requirement* refers to at least one of the requirements coming from the scenarios in D4.1 [22] and summarized above. To evaluate according to this requirement(s), the function or system needs to perform some *Actions*, which will bring the system into another, well-defined state that can be evaluated according to a simple or complex *Metric*. Metrics are especially needed if the evaluation is done by simulation – the operation of an algorithm will result in gain, which can be quantified. Finally the last section contains the *Evaluation* itself and can

consist of an analysis or a simulation. Some of the evaluations are already presented in papers and other 4WARD deliverables, so this section might include a short summary of the evaluation with a reference to a more detailed description.

4.4 Demonstrator

An additional aspect of an evaluation is the transfer of the design concepts to a prototype. Deliverable D-4.4 [24] reports in detail experience gained by WP4 in the implementation of a selected set of INM functions and its realization within realistic scenarios for the future Internet.

The prototype shows the feasibility of the distributed architecture and reports details about interfaces for implementing it over a real testbed. The adaptive properties of the INM algorithms are mapped to an objective interfaces that shows their use inoperative networks. The graphical interface shows also how the aggregated metrics created by the INM algorithms can increase scalability in the management information as well usability in management operations.

The prototype has been developed following a reference scenario. This allowed the definition of several use case for INM in conjunction with other WPs in 4WARD. The reference scenario is shown in Figure 4-5 and is built on the requirement of managing quality of service under changing conditions. The results of the prototype activity are reported in D-4.4 [25].

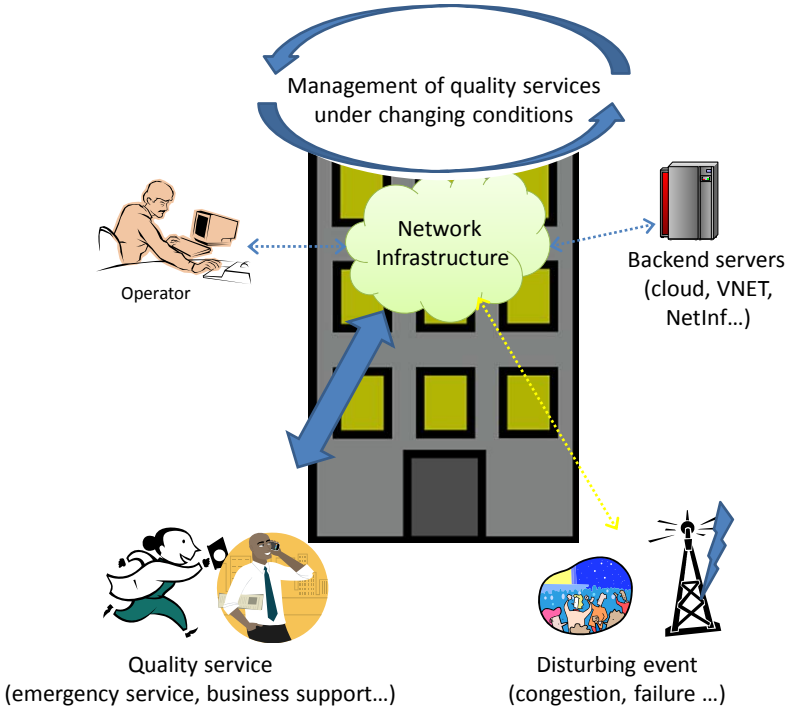


Figure 4-5: Reference scenario used for the integrated prototype.

5 Evaluation of the Framework

Before providing detailed evaluation of INM algorithms based on the template of section 4.3.1, we present here an analysis of the benefits of the INM framework. The discussion is presented according to the following structure:

- ***5.1.X Requirement***
  Description of the Requirement
- *Assumptions:*
  Preconditions and description of environment
- *Evaluation of Requirements:*
  The requirement is evaluated against use cases.

5.1 INM Framework Overview

The INM framework has been extensively presented in D-4.3 [24] on its design aspects, namely architectural elements and operations between interfaces. The main objective of the framework is to support INM algorithms in real deployments: it defines which platform can be used, how algorithms communicate with each other and how network can be operated.

Existing frameworks for network management fall short in a few requirements for managing the future Internet: for example complex interfaces to access management information, a fixed top-down structure, and lack of adaptability are common limitations.

The INM framework has instead introduced a new architecture and methods to deploy advanced management algorithms. The impact of the INM framework focuses on two major areas: software design of management capabilities and properties of distributed systems. To substantiate the benefits of the INM framework, we report an evaluation study on four key properties of enhanced network management:

- scalability of management operations
- robustness
- reduction of integration effort
- reduction of complexity

Some of the properties are strictly related to network operations, while others (especially the reduction of integration effort) are relative to software integration. We focus our discussion of the INM framework on the analysis of realistic exploitation plans and we present concrete results on specific use cases. An analysis based on numerical evaluation is used instead mainly in section 6 for the INM algorithms.

The purpose is to give a convincing assessment of the INM framework as instrument to deploy INM algorithms in realistic exploitation scenarios.

5.1.1 Scalability of management operations

Large and dynamic networks pose several challenges to their management systems. These challenges can be generally considered as related to their scalability. The difficulty of applying management functions to a large set of nodes, especially in those situations where a function cannot be merely repeated over those nodes without any adaptation is one example of one of these challenges. For example, dynamic changes in the requested traffic or load distribution require quick monitoring and control of network resources, which are not trivial to enforce on the large scale.

The technical issues related to the execution of a function to different nodes should not be underestimated. A first method to increase scalability of management function is to aggregate management information across different nodes, but this can be performed –in its basic implementation- only across similar network functions. Correlation of information across access and core network requires for example advanced aggregation functions. Scalability in the future Internet is therefore a requirement applied not only to the dimension of the networks, but also to the complexity of the networks functions there deployed, such as technology, topology, performance requirements at service level, layers implemented.

These challenges are in general referred to as diversity in the network, which can be with respect of technology, topology, performance requirements at service level, layers implemented as well as the scale of the network. Scalability in the future Internet is therefore a requirement applied not only to the dimension of the networks, but also to the complexity of the networks functions there deployed.

The constructs introduced in the INM framework enable the composition of monitoring and adaptation across different nodes and functions; few use cases of self-management are here reported as examples. The impact of INM on specific performances in the network (traffic generated, timeliness etc.) depends on the algorithm implemented and are instead discussed more in detail in the next chapters.

Assumptions

One of the basic concepts of the Self Managing Entity (SE) is that management capabilities are co-designed with the network function, and therefore a local mapping between low-level parameters and organization interface has taken place. Application of the co-design principle requires an understanding of the network function and the ability to abstract and design a high level interface (i.e. the organisation interface).

Evaluation of requirement

Scalability can be evaluated on different cases. The first one is the construction of aggregated information through the composition of objectives across different areas. The INM framework introduces aggregation through the concept of domains, which define the set of nodes in which the objectives are aggregated. For example, in traditional networks the construction of KPIs requires much off-line processing, where a topology database is used to extract values coming from a selected set of nodes. INM algorithms for real-time monitoring, instead, build KPIs in a distributed manner and the domain is used to select the set of nodes for composition; no additional filtering function is required for processing the data gathered.

A full validation of this approach as instrument to increase scalability – especially in real-time operations- requires a full specification of interfaces for the INM objectives, and is therefore further described in D-4.4 as part of the INM prototype.

The INM Framework defines the Organisation interface as the basic building block of a hierarchical distribution of management functionality in a system. The number of levels in the hierarchy is left unspecified in the framework so that implementations can be optimized for different systems and technologies.

The INM Framework supports delegation of objectives across different layers, so that management functions can be performed locally where the network functions is located. The information exchanged between different levels of the hierarchy is specified in Service Level Agreement contracts associated with the composition of SEs via the Organisation interface. Through this approach not all the amount of information generated in real-time is reported to the operator, but INM algorithms can filter it locally for distributed monitoring and adaptation.

A second case where INM needs to prove scalable is operation of networks containing different transmission technology. An example can be dense mobile networks, where the network needs to operate integrally between the radio access section and the backbone section. Here the INM algorithms need to be composed not only across similar nodes (e.g. to create KPIs related to radio technology), but they need to cooperate across different metrics (e.g. quality metrics of radio channels and congestion metrics of fixed networks).

With this respect the organization interfaces provides a means to clearly identify the different sections of the network, to extract the relevant metrics from each of them and to compose management capabilities for self-optimization. Figure 5-1 shows a use case that has been evaluated in 4WARD in the context of physical layer awareness for heterogeneous networks (see [26] for a complete description).

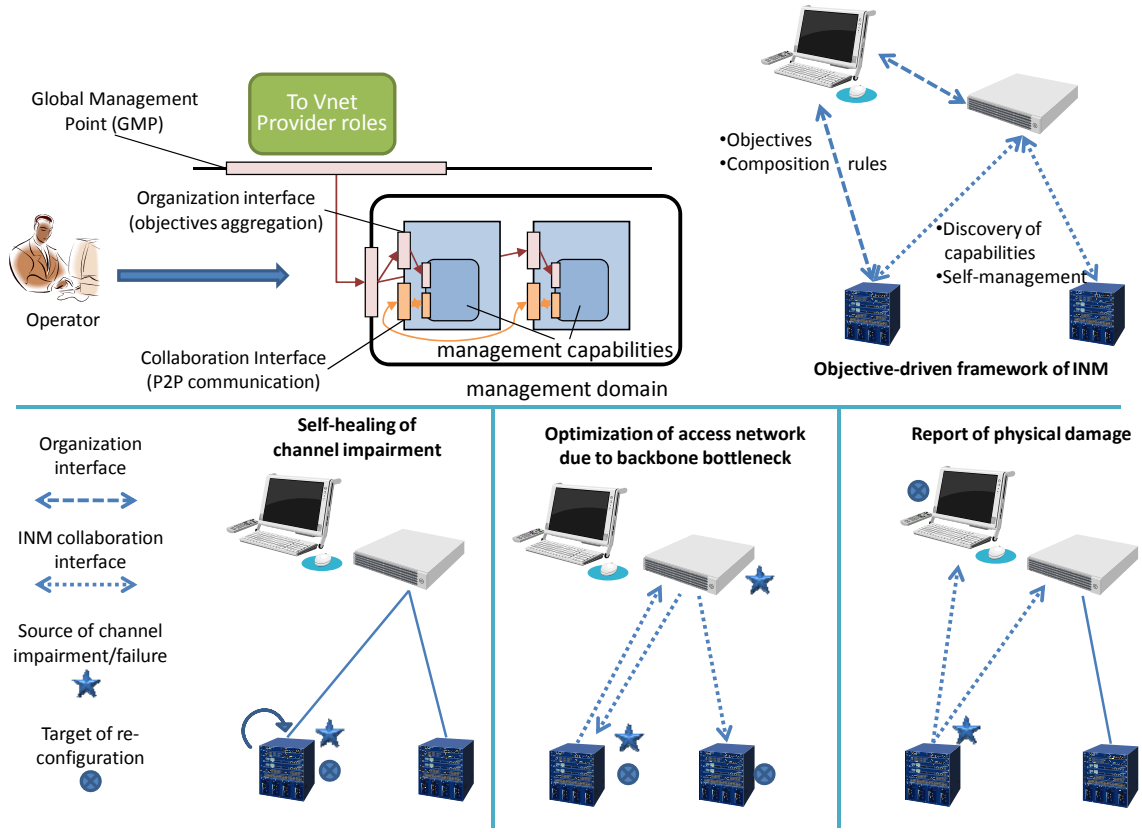


Figure 5-1: Use case for heterogeneous networks

5.1.2 Robustness

Robustness is another important feature supported by the INM framework for managing future networks. Robustness is concerned with maintaining consistency of performances of the management system under different conditions. What happens if a node fails? How accurate is the real-time view of a wireless network in case of link disturbances?

Traditional network management guaranteed a certain level of robustness through redundant machines and off-line operations. These instruments will not be adequate in the future for scalability aspects (see previous section 5.1.1) and for cost saving reasons.

Distributed architectures supports better scalability, but their reliability under different conditions is still an issue for operative networks. In fact characteristics like traffic overhead

and topology structures are quite sensitive to changes in the network environment and they might impact the robustness of the network management function if not considered.

The INM framework includes the interfaces to control these characteristics, in such a way that the cost and behaviour introduced by distributed algorithms can be integrated into a set of predictable management functions. (For example anomaly detection can work differently depending on the technology being used).

When moving towards an INM environment, the co-design principle is central to success. The design of the INM framework is underpinned by this principle, by which the entity which needs to be managed and the management logic which performs this task should be designed in parallel. The Self Managing Entity is an encapsulation of a network function that needs management and the management logic (in the form of Management Capabilities) which carry out the task.

Assumptions

Applying co-design to its full extent would mean building some entity or service from scratch. The framework assumes that algorithms are designed accordingly to an analysis of properties of distributed algorithms, namely accuracy, overhead, cost. In addition to that, these parameters are made available to the organization interface.

Evaluation

If the INM framework is used correctly it will apply the co-design principle as far as is possible. If this is achieved then an increase in reliability would be expected. The reasoning behind this is that with management co-designed with a service, management becomes more efficient as the service itself has allowed for management in the most appropriate form.

The INM Framework underpins the SE and the properties which the SE exhibits. Several of these properties are central to system robustness: self-monitoring and self-diagnosis. In addition, through decentralization which the INM Framework supports (through the collaboration interface), the SE is allowed to take decisions to act and repair any defects that are within the scope of its management capabilities. It should be noted that the benefits of this approach would be available even in a migration scenario from the current network, where only part of the network functionality would be implemented natively as SEs while the majority of the functionality would be based on existing technology managed by dedicated management entities that interact with the SEs.

The algorithms proposed in deliverable D-4.3 consider key properties of distributed systems (e.g. timeliness and overhead) and the SE support the interfaces to control these properties; the use of these interfaces is explained in D-4.4 and is briefly depicted in Figure 5-2.

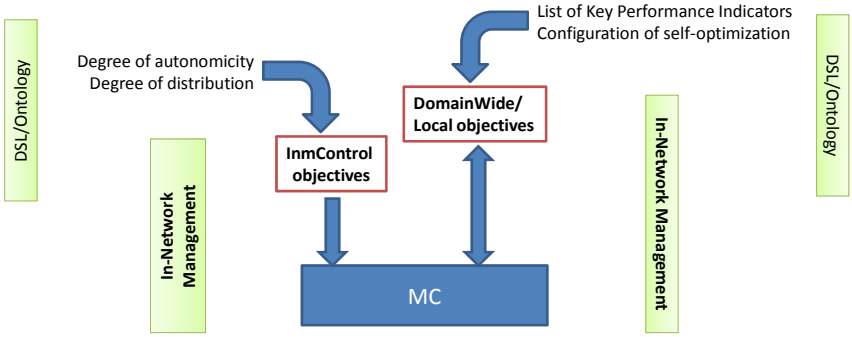


Figure 5-2: Robust control of INM behaviour

The configuration of these properties allows changing the reliability of self-management functions and therefore changes the level of robustness achieved in the network. For example the frequency of probe messages used by the anomaly detection determines the delay by which faults are discovered in the network. Networks with more dynamic performances (e.g. mobile radio networks) can be therefore made more or less robust based on the configuration allowed by the INM framework.

5.1.3 Reduced integration effort

Promoting the development and deployment of management algorithms inside the INM framework is of paramount importance. If this is to be achieved, the development process must be intuitive and not constraining on algorithm developers.

The major benefit for network management is the impact on the integration effort between the managed function and the management capabilities. Traditionally, developers of management functions are required to carefully analyze the interface and parameters of installed functions and add management capabilities on top of them. The approach of the INM framework proposes to integrate management functions in the design of services, so that much of the mapping is performed inside the implemented logic.

The framework allows for positioning management, in the form of Management Capabilities, at different levels, namely:

- Inherent – very tightly coupled with the entity being managed, e.g. TCP flow control mechanisms in today's networks. The management is part of the protocol.
- Integrated – coupled with the entity being managed, e.g. ANR (Automatic Neighbour Relation) functionality in LTE. This detects and configures the relationships between cells. It is part of the cell management.
- Separated – decoupled with the entity being managed, e.g. a monitoring algorithm which is not specific to any service but which a service can make use of. The GAP algorithm is an example of this in that it provides a monitoring of whatever parameters are needed
- External – completely external, more so non INM management, e.g. the OSS functionality today is external to the network itself. Inter-domain management may still need to reside at this level.

As example, we introduced in [24] the Cross-Layer QoS (CLQ). To allow high level entities to take optimal decision, each physical node runs dedicated software to monitor low level parameters, as well as to impose a specific behaviour. The Cross-Layer QoS (CLQ) capability included within the INM framework is able to measure and to control the following set of parameters: One-Way Delay (OWD) and respectively Available Transfer Rate (ATR) with the neighbouring nodes.

Assumptions

For the top-down approach, it is supposed that CLQ is able to address requests to the hardware (i.e. Physical Layer and MAC Sub-Layer), in order to change the infrastructure behaviour. These requests are issued by managers or high level management entities. Another assumption refers to the presence of a collaboration interface with the hardware for bottom-up approach. For an automatic operating mode, a discovery management capability is needed.

Evaluation of Requirements

A first aspect to consider is the feasibility of the approach in real implementation of distributed algorithms. For this purpose, the INM framework puts a very limited number of constraints on developers:

- Encapsulate the algorithm logic inside one or more Management Capability (MC) constructs. The number of MCs is left up to the algorithm developer as they know best the inter-workings of the algorithm. It is a modular approach.
- Management Capabilities must implement two interfaces, an Organisation and Collaboration. The Organisation is a north bound interface where configuration changes can be applied. The Collaboration interface is an east west bound interface which is used for collaborating with peer MCs. It is expected that most of the functionality available on these interfaces will have to be standardised, while allowing for proprietary extensions [23]. One proposal, which was not implemented during the project lifetime due to time constraints, was to adopt a service-oriented approach based on REST principles [24]. This would result in simple definitions of the interfaces themselves and of the operations associated to them, with obvious effects on the complexity of the integration as long as the proprietary extensions are limited.
- The algorithm must publish its key outputs through a management objective, e.g. an algorithm monitoring congestion should publish this congestion level. This objective can then be subscribed to by other system entities. For a developer it is just a publish to the INM CORE and the INM CORE handles all of the subscriptions.

As proof of concept, a number of algorithms have been implemented and deployed inside the INM framework, like Generic Aggregation Protocol (GAP), Congestion Control, QoS Monitoring, Anomaly Detection (see D-4.4). All these algorithms are very distributed in their communication paradigm, but they are commonly controlled through the objectives of the organization interface.

For example, INM CLQ is providing real time information regarding neighbouring nodes and the communication links. Based on this low level network status, a composite metric is automatically calculated to offer an overall perspective of available resources. Cross-Layer QoS offers the information used by other INM capabilities. For instance this is used by DSL to produce high-level KPIs. Vice-versa, other algorithms or applications could address request against CLQ, “translated” to be understood by the directly monitored hardware. This low level information can be offered periodically at a specific time interval or by request.

Of particular interest is the fact that joint development of some QoS features between the INM framework and the GP framework. In fact, traditionally real-time optimization of the network infrastructure is deployed as separated functions in the control plane. In 4WARD many of these traditional functions have been jointly developed as party of INM and part of GP. As output, important QoS features like congestion control or link failures are co-designed as management functions and implement the INM interfaces. In real settings, this can be seen as a clear reduction in the integration effort between deployment of infrastructure and addition of management functions.

Considering QoS in detail, once the neighbours are discovered, the CLQ capability instantiated in any physical strategic node (i.e. a node implementing GP, NetInf and major management capabilities such as neighbour discovery, registry, resource control, event handling, security etc.) has self-configuration characteristics. If the nodes do not implement the above mentioned functions (very likely for the legacy devices nowadays), but they have minimal CLQ functionalities (measurement part at least), they could interact according to the new paradigm through a strategic node.

Finally, the architecture of INM has an impact on migration aspects, related to the shift of centralised external management solutions to more distributed and autonomous. It allows for redeployment, in a modular fashion, of management logic into the network. The framework allows for an optimal positioning of the management logic being redeployed and doesn't enforce inherent management but of course does promote inherent management where appropriate. An example of migrating existing legacy management functionality is as follows; take the existing monitoring of node status(alive or dead) from an OSS in a RAN today. Some existing solutions use periodic pings from the OSS to each node. A simple Management Capability could be deployed on all nodes and it executes a heartbeat message back towards the OSS. This level of embedding of this MC would be separated.

5.1.4 Reduction of complexity

Complexity in operating large networks is a well-known issue in today's networks. Besides being a source for higher operational cost, complexity is also an obstacle to make changes in the network and therefore also to introduce new services in the network. Complexity has never been reported as a quantitative property of a network, but it comes as results of experience when integrating different functions of the network and operating them consistently.

The role of a network management framework with this respect is to simplify operations within the network and reduce the amount of information required to operate the network, while at the same time guarantee the same level of reliability.

Assumptions

There are no assumptions made for this requirement.

Evaluation of Requirements

A first instrument to reduce complexity is simple constructs to build operations for network management. An important element here is the definition of adequate interfaces that allow control of management capabilities, but at the same time limit to the minimum the amount of mapping between different operations.

The INM framework has introduced the organization interface as means to abstract some of the complexity in managing network functions. This interface exposes only high-level objectives that are aggregated from the detailed configuration parameters of the network function. For example an INM objective constructed through this interface is a health status of a router, which aggregates the status of link interfaces, congestion control etc. This objective can be used by real-time monitoring as well as trigger to initiate self-optimization in the network. Much of the mapping between the INM high level objective and the internal logic of the algorithm is performed in the form of internal logic of the management capabilities.

Following this principle of information hiding, the INM framework is based on the definition of a set of objectives as real-time indicators. A disturbance in the network is normally not reported through the INM interfaces, but it is instead handled internally through triggering of other INM capabilities. For example, congestion is not monitored directly, but a correlated performance indicator is created in the INM framework (delay, bandwidth available); in case congestion occurs, an internal event is generated to trigger path reconfiguration (see D-4.4), but no report is sent directly to the network operator.

Another important instrument to reduce complexity is the software tools used for INM. Since networks are changing rapidly in the type of services deployed, transmission media and usage by users, it is often difficult for an operator to put in place a complete set of management functions: gradual deployment of these functions through remote upgrades is becoming instead an important requirement for efficiency in operating next generation networks.

The model of INM capabilities has been designed in compliance to the OSGi platform that is becoming a standard for remote control of network elements. The organization interface is a single point of attachment for INM capabilities and can be mapped, in a straight forward manner to the interfaces of services. In this way, a management capability can first of all be dynamically deployed in the network in case a new function needs to be added (for example, a new QoS classes is added to the network router, then a new mapping function is required to build the high level objectives). In addition, the INM framework can then rely on a set of basic functions supported by the platform, for example discovery of interfaces across nodes.

A Domain Specific Language has been developed as part of the framework (ref D4.4) which can be used to gather information from deployed Management Capabilities (e.g. an algorithm). The DSL provides operations which can be used to compare, contrast and abstract higher level knowledge from the underlying Capabilities in an open and intuitive fashion.

The DSL can be used for potential composition of algorithms, which again will reduce complexity. A script written with the DSL can listens for congestion updates from an aggregation algorithm (e.g. GAP) and in turn trigger the startup of a congestion control algorithm which will attempt self healing (e.g. congestion control algorithm) when a certain congestion level is reached. Using the DSL in this fashion ensures that both algorithms can be developed and deployed independent of each other. Their composition can occur at runtime.

Co-Design of Management and Service Functions

Reducing the complexity of large communication systems in general is also achieved by well-known principles such as modularity, layering, hierarchies, and various forms of interaction (e.g. cross-layering). Complementary, design patterns have emerged to facilitate the implementation of large software systems [32] [33]. While some principles are common practice today also in network management, such as SNMP's hierarchical management structures, lack of support for the structured design of embedded management processes persists. We argue in [34] that integrative aspects of management and service realms should be exploited in the design of management solutions, because in many cases, both knowledge and functions for realizing management tasks are shared between both realms.

To this end, we introduce co-design patterns in [35] to network management that support in the design of embedded, distributed, and large-scale management systems and thereby aid in the reduction of complexity. In general, the concept of co-design originates from the very observation that knowledge and functionality about how to manage a system is typically split between multiple roles involved in the operation and management of the target system. For instance, service designers and network operators may well know how to manage different aspects of a service and also have expertise in integrating them into the backend management operations. Co-design patterns proactively support the exploitation of synergies between such parties: they represent a set of structural blueprints of how to construct parts of a management system by combining knowledge and functionality of different parties to facilitate the reuse of existing functionality, to simplify management function design, and to increase system performance.

Figure 5-3 illustrates the relevant concepts of in-network management (INM) where co-design patterns apply. Figure 5-3 (left) sketches the structure of a network node, where both service and management logic are integrated into a single coherent, deployable self-managing entity. Management functions are invoked by calls to management capabilities, which implement algorithms that realize the management functions, such as fault handling. Typically, a management control loop is formed by multiple interacting management capabilities of self-managing entities that span several nodes in a communication network (Figure 5-3 (right)). Invocation of management capabilities by service processes and vice versa is performed, for instance, by function calls, and control between both sides is transferred accordingly.

Supporting in the design of interactions between embedded management and service processes is the objective of the proposed co-design patterns.

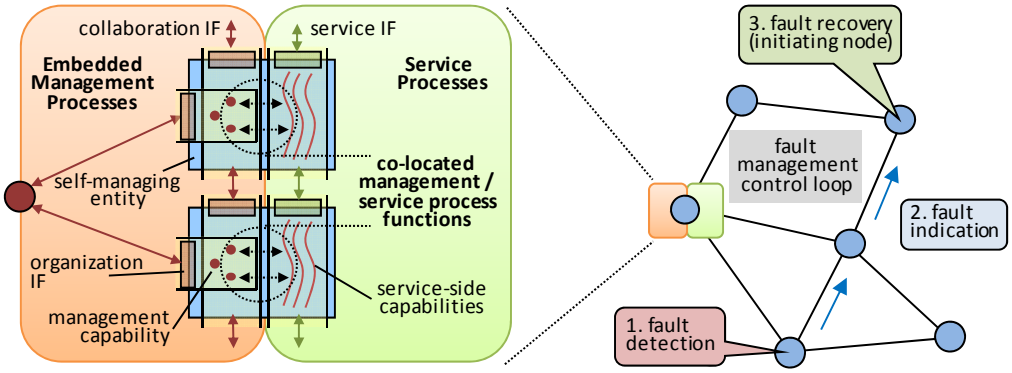


Figure 5-3: INM-compliant node structure (left) and distributed fault management control loop (right).

In [35], we propose an initial nonexhaustive set of co-design patterns for in-network management that model typical recurring problems in the fine-granular interactions between management and service functions. The simplest of a number of patterns, *control handover*, makes explicit that control is handed between service process and management process in either direction. This pattern separates both spaces in functional terms and helps in understanding the separation of concerns in the design phase of complex management systems involving many management capabilities, self-managing entities, and network elements. This pattern applies, for example, to the situation of a service-side security exception that leads to the invocation of a security-related management capability.

In order to demonstrate the power of co-designed in-network management solutions, [35] contains an evaluation of an ns-2 based simulation of a complete data migration suite in MANETs. In the evaluation, a co-designed solution of the management control loop illustrated in Figure 5-3 (right) is compared to a corresponding non-co-designed solution. While in principle, the co-designed solution can also be implemented without making use of co-design patterns, applying suitable patterns can substantially reduce overall complexity because functional synergies between management and control functions can be more easily exploited. Furthermore, identifying co-design patterns during system design also aids significantly in the understanding of a complex management system.

The following two figures compare the performance of the co-designed and non-co-designed realization of the management control loop of Figure 5-3 (right). The mean fault recovery time is shown in Figure 5-4 (left) as a function of the checking interval, a configurable parameter in the implementation that indicates how frequently a fault check occurs. For the three partition duration intervals shown, the co-designed solution clearly outperforms the non-co-designed one in every case. The figure suggests further choosing small checking intervals to improve mean recovery time. Figure 5-4 (right) shows three graphs with the communication cost required for executing the management control loop in Figure 5-3 (right). In all cases, the co-designed solution has superior performance over the non-co-designed solution. What's more, choosing small checking intervals is not supported by this figure due to the significant increase in communication cost (note the logarithmic scale on the y-axis). Hence, Figure 5-4 makes clear that the non-co-designed solution is not adequate in either case, whereas the co-designed solution performs efficient and even constant in the considered settings.

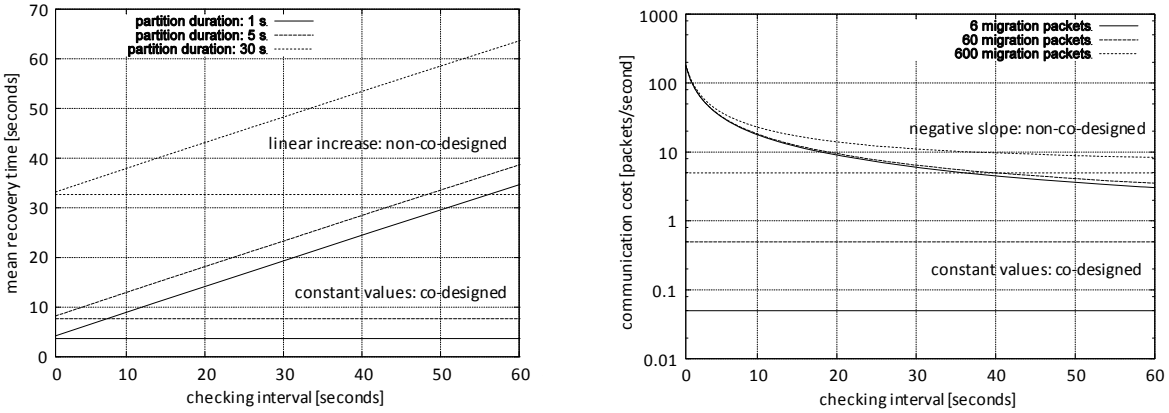


Figure 5-4: Mean fault recovery time (left) and communication cost (right)

6 Evaluation of Distributed Management Algorithms

This chapter evaluates specific distributed management algorithms that have been developed during the last 30 months in WP4. Section 6.1 addresses the algorithms of INM Situation Awareness and Section 6.2 analyzes the Self-Adaptation behaviour of developed algorithms.

Analogue to the same arguments described for the evaluation of the INM framework 5 we use running text rather than fully applying the previously introduced evaluation template.

Here is a mapping for the algorithms from the presented evaluation form.

> **6.x.y *Name***
> Description of the algorithm
>
> *Assumptions:*
> Preconditions and description of environment
>
> *Evaluation of Requirements:*
> Similar requirements are grouped. Instruments are analysis and simulation. This section also includes *actions* and *metrics* if simulations are used.

It is important to notice that following algorithms deal with very specific INM problem spaces and thus also address very specific functional requirements to INM rather than general ones. Table 6-1 summarizes and visualizes for each distributed management algorithm evaluated in 6.1 and 6.2 the functional requirements that are addressed. Vice-versa the table also helps to pick out a certain requirement and to quickly figure out by which algorithm it is addressed. If a requirement is addressed it is indicated as **x**, if not it is indicated as **-**.

An overall conclusion of the requirements that are addressed is described in 9.

Requirement \ Section	6.1.1	6.1.2	6.1.3	6.1.4	6.1.5	6.1.6	6.1.7	6.1.8	6.1.9	6.1.10	6.2.1	6.2.2	6.2.3	6.2.4	6.2.5	6.2.6	6.2.7	6.2.8
Information Gathering and Collection																		
Monitor of lower layer information	-	-	-	-	X	-	X	X	-	X	-	X	-	-	-	X	X	X
Situation awareness, detection of network conditions	X	X	X	-	X	-	X	X	X	X	-	-	X	X	X	-	X	X
Detection of network anomalies	-	-	X	-	X	-	-	X	-	-	-	-	X	X	-	-	-	-
Characteristics of devices	X	X	-	-	X	X	-	X	-	-	-	-	-	-	X	X	-	-
Information about network resources	X	X	-	-	X	X	X	X	X	X	-	X	X	X	-	X	X	X
Information Distribution and Node Collaboration (by Node Interaction)																		
Common Information model and protocols	-	-	-	-	-	-	X	-	X	-	X	-	-	X	-	-	X	X
Distribution of captured and collected information	-	-	-	-	-	X	X	X	X	X	X	-	-	-	X	X	X	X
Multicasting status and capabilities of nodes	-	-	-	-	X	-	-	-	-	-	-	-	-	-	-	-	X	-
Information exchange done in a standardized way	-	-	-	-	-	X	X	-	X	-	X	-	X	X	-	X	-	-
Distributed management and role based interaction	X	X	X	X	-	X	X	X	-	-	X	-	X	-	X	X	-	X
Security, establishment of trust relationships	-	-	-	X	-	-	-	-	X	-	-	-	-	X	-	-	-	-
Core Network Management Requirements																		
Management based on situation, policies and/or business objectives	-	-	X	-	-	-	-	X	X	-	X	-	-	X	X	-	-	X
Self-Management e.g. according to FCAPS model	X	-	-	-	-	-	-	X	-	-	X	X	-	-	-	-	X	X
Self-Adaptation of network components	X	X	-	-	X	-	-	X	X	-	X	-	-	X	-	X	-	X
Distributed Network Management architecture	-	-	-	X	-	-	X	-	-	-	X	X	X	-	X	X	X	-
Make decisions and take actions	-	-	-	-	-	-	X	X	X	-	X	-	-	-	X	X	X	X
Special, Data Communication Related Requirements																		
Traffic differentiation and handling	-	-	-	-	-	-	-	X	X	-	-	-	-	-	-	-	-	-
Routing related functionality: Route discovery or self-rerouting	-	-	-	-	X	-	X	X	-	X	-	-	-	-	-	-	-	X
Performance of Network Management Mechanisms																		
Distributed Management reduce computation load on each node	X	X	X	X	-	-	X	-	-	-	X	X	-	-	X	-	X	-
Distributed Management reduce management information flow	X	X	X	X	X	X	X	-	-	-	X	X	X	-	X	-	X	-
Reliability at least similar to centralized network management approach	X	X	X	-	-	-	X	-	-	-	X	X	X	-	-	-	X	-
Substantially faster adaptation compared to centralized approach	-	-	X	X	-	-	X	-	-	-	X	-	X	-	-	-	-	-
Quick switching of network wide behaviour	-	-	-	X	X	-	X	X	-	-	X	-	-	-	X	X	-	X
Self-adaptive behaviour optimizes available resources	X	X	-	-	X	-	-	X	X	X	X	-	-	X	X	X	-	X

Table 6-1: Overview of addressed functional requirements by distributed management algorithms

6.1 INM Situation Awareness

Situation awareness consists in estimating the state of the network system. This is a very wide field and, clearly, we can not cover it completely. We have provided a selected set of functions that we regard as important and challenging for the management of the future Internet. The result is a set of complementary distributed algorithms that provide views of the network state in real-time. This functionality includes real-time monitoring of network-wide metrics, group size estimation, topology discovery, data search, anomaly detection, distributed reputation aggregation and wireless path quality prediction. These algorithms provide the necessary input to the self-adaptation mechanisms.

Regarding the **monitoring of network-wide metrics**, we have developed solutions for real-time monitoring of network-wide metrics, such as average or peak load (Section 6.1.1). We have also developed solutions to detect a threshold crossing of a network-wide metric (Section 6.1.3), indicating a problem that may need attention. Our solutions are based on both tree-based and gossip-based underlying protocols.

Also in the context of the monitoring of network-wide metrics, we have developed stochastic models of tree-based aggregation under churn (Section 6.1.2), i.e., where network nodes may be dynamically removed from, or join, the network. We have developed several performance models.

Finally, we have developed secure versions of our algorithms for monitoring of network-wide metrics (Section 6.1.4). These versions are able to execute without providers private information leaking to outsiders. This is particularly important for network management information, as this generally contains lots of information about the configuration, operation, load, and performance, of the providers' internal network.

For providing **group size estimation**, we have engineered NATO! (Section 6.1.5), a statistical probability scheme for estimating the size of a group of nodes affected by the same event without explicit notification from each node, thereby avoiding feedback implosion. An efficient solution for this task permits, for instance, monitoring the operating conditions of a large-scale network by computing the share of nodes whose performance is above (or below) a given threshold. Note that NATO! provides one type of network-wide metrics (i.e., COUNT). It implements an alternative approach to that of the solutions in Sections 6.1.1-6.1.4. NATO! is based on suppressing the messages sent by monitoring agents towards a management station. In contrast, the other solutions are based on aggregating messages inside the network.

We have engineered “Hide and Seek” (H&S) (Section 6.1.6), a novel algorithm for **topology discovery**. In highly dynamic scenarios, like the ones we target, the need for efficient topology discovery is particularly important.

In the context of **data search**, we have investigated the efficiency of random walks and flooding for exploring networks, based on case studies evaluated by simulation and transient analysis (Section 6.1.7). Specifically, we have considered, single random walks, multiple random walks, biased random walks and flooding.

We have developed a distributed approach to adaptive **anomaly detection** and collaborative fault-localisation (Section 6.1.8). Clearly, the relevance of this management task is higher in the dynamic future Internet than in traditional, more stable, scenarios. Note that the solution for detecting threshold crossings of network-wide metrics (Section 6.13) can also be used for detecting anomalies at the network-wide level, while this work focuses on the device-level.

Moreover, we investigated the impact of **distributed reputation aggregation** as enabler to avoid network overload (Section 6.1.9). This system can be especially of use to protect systems from attacks and thus keep services available to users.

Finally, a cross-layer approach to **predict the path quality** in **wireless** mesh networks has been developed which is separated from the routing logic (Section 6.1.10). It also supports the use of multiple routing protocols simultaneously.

6.1.1 Continuous Monitoring with Performance Objectives

A-GAP is a monitoring algorithm that provides a management station with a continuous estimate of a global metric for given performance objectives. A global metric denotes the result of computing a multivariate function (e.g., SUM, AVERAGE and MAX) whose variables are local metrics from nodes across the networked system (e.g., device counters or local protocol states). Examples of global metrics in the context of the Internet are the total number of VoIP flows in a domain and the list of the 50 subscribers with the longest end-to-end delay.

Our approach is based on in-network aggregation, where global metrics are incrementally computed using spanning trees. Performance objectives are achieved through filtering updates to local metrics that are sent along that tree. A key part in the design is a model for the distributed monitoring process that relates performance metrics to parameters that tune the behaviour of a monitoring protocol. The model allows us to describe the behaviour of individual nodes in the spanning tree in their steady state. The model has been instrumental in designing a monitoring protocol that is controllable.

Assumptions

- There exists a distributed management architecture, whereby each node in the networked system participates in the monitoring task by running a monitoring process, either internally or on an external, associated device.
- Local metrics can be accessed on each node, where they are periodically updated in an asynchronous fashion.

Evaluation of Requirements

Information Gathering and Collection

- Situation awareness, detection of network conditions
- Characteristics of devices
- Information about network resources

A-GAP provides a continuous estimation of global metrics in real-time. This is, it provides a key building block in the creation of situation awareness in real-time. The semantics of the metric is transparent to A-GAP, which can track different metrics related to network devices. The output of A-GAP can be directly consumed by algorithms that perform anomaly detection.

Information Distribution and Node Collaboration (by Node Interaction)

- Distributed management and role based interaction

A-GAP is a distributed algorithm where each node in the networked system participates in the monitoring task by running a monitoring process. More specifically, in A-GAP, all nodes solve (independently and asynchronously) a different instance of the same optimization problem.

Core Network Management Requirements

- Self-Management e.g. according to FCAPS model
- Self-Adaptation of network components

A-GAP is autonomic (self-*) in the sense that given a set of performance objectives, such as the quality of the estimation, it configures itself in a way that these objectives can be met, and it dynamically adapts to changing conditions.

Performance of Network Management Mechanisms

- Reduction of computation load on each node by using distributed management
- Reduction of management information flow by using distributed management
- Reliability at least similar to centralized network management approach
- Self-adaptive behaviour optimizes available resources

Our evaluation results [19] show that we can effectively control the trade-off between accuracy and protocol overhead, and that the overhead can be reduced significantly by allowing small errors. A lower protocol overhead means that each node has to process fewer messages and therefore the computational load on each node is reduced.

The distributed nature of A-GAP makes it resilient to node failures. Furthermore, it adapts very quickly. For instance, for all simulation scenarios we simulated, in case of a node failure, the adaptation time is very short: the settling time for the accuracy is a fraction of a second, and it takes a few seconds for the overhead to settle. The algorithm also provides, in real-time, an accurate estimation of the adaptation time to a change.

A-GAP continuously self-configures to provide the global metric with the required accuracy and minimal overhead, reducing the resource consumption.

6.1.2 Aggregation Under Churn

We consider a restricted variant of GAP [16], a tree-based aggregation protocol, which continuously computes the node count in a network under churn. The protocol is self-stabilizing in the sense that, once no churn occurs, the network and aggregation data eventually become static, the overlay forms a stable breadth-first search tree, and the root node has the correct count.

The protocol's behaviour in a network over time can be described using a continuous-time Markov model parameterised on the Poissonian node join and failure rates λ_j and λ_f, where $\lambda_j = N\lambda_f$, with N the initial number of nodes. This means that in expectation, the network size, i.e. the number of nodes in the graph, is N [27]. Under simplified assumptions, it is then possible to calculate the expected accuracy of the count at the root node, given the node protocol cycle rate λ_g, which is the rate at which a node samples its neighbours' partial aggregates, updates its state, and sends its aggregate to a parent node. Specifically, in the protocol, every node keeps track of its current level, which is its current belief in the number of hops needed to reach the root node. This allows defining two families of stochastic variables N_x and M_x, which estimate the number of nodes with level assignment x and the aggregate held by a node at level x, respectively. Then, $A_x = M_x N_x$ estimates the total aggregate at level x. In the analysis, an equation with A_x/N as left-hand side, allowing numerical solution, is given.

Simulations have shown that errors in model predictions of accuracy are small for large values of $r = \lambda_g/\lambda_f$. As illustration, Figure 6-1 shows simulations of the normalized aggregate A_x/N as a function of the level x for churn rate r, and random network of expected node degree (average number of neighbours) k, for networks of size 10^3, 10^4, and 10^5, respectively. The figure shows an excellent fit between model and simulation. Further details are available in [24] and [17].

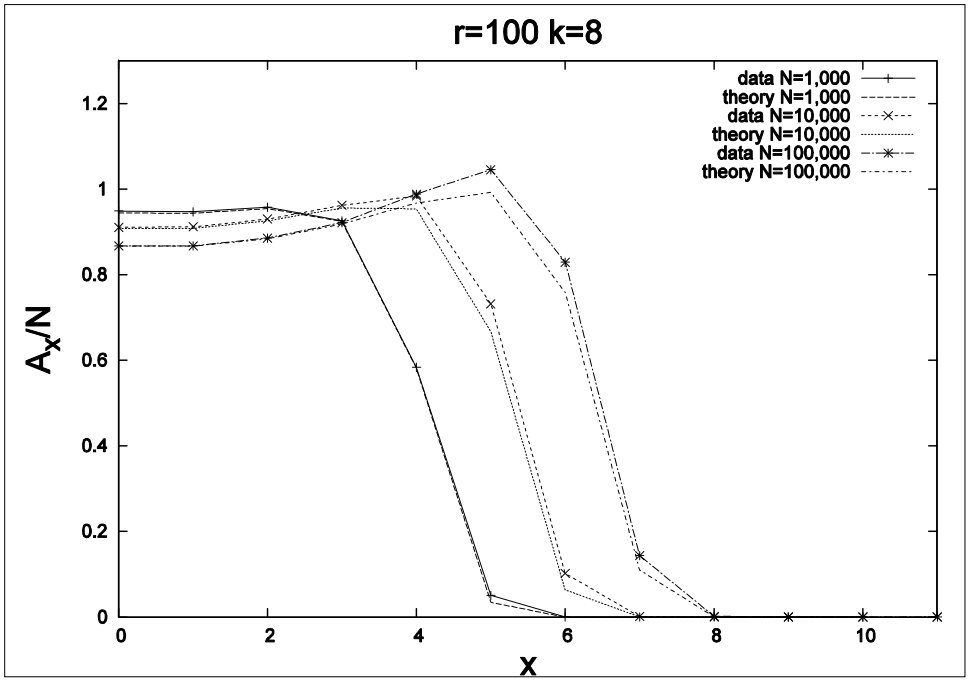


Figure 6-1: Normalized aggregate A_x/N as function of the level x for churn rate r

Assumptions

- The network is an Erdös-Rényi random graph with N nodes initially.
- The global node join and failure rates, λj and λf, are Poissonian, i.e. the number of joining and failing nodes in a time interval of length t are Poisson distributed random variables with parameters λjt and λft.
- Each active node undergoes a protocol cycle with a Poissonian rate λg.
- A joining node has an a priori Poissonian degree distribution and attaches itself to already present nodes so that any allowed set of neighbours is equally probable.
- When a failure occurs, all active nodes are equally likely to be affected, with the selected node being removed from the graph.

Evaluation of Requirements

Information Gathering and Collection

- Situation awareness, detection of network conditions
- Characteristics of devices
- Information about network resources

When run on a random network graph which undergoes failures and joins constantly, the GAP variant estimates the number of nodes in real-time, creating situation awareness. The output can be directly consumed by algorithms that take current network size as input, such as anomaly detection algorithms.

Information Distribution and Node Collaboration (by Node Interaction)

- Distributed management and role based interaction

The GAP variant is a distributed algorithm where each active node in the networked system participates in the counting task by sampling its neighbourhood and propagating local information upwards in a tree overlay.

Core Network Management Requirements

- Self-Adaptation of network components

The GAP variant takes into account changes in topology resulting from joining and failure of nodes. It inherits the self-stabilizing traits of the breadth-first search algorithm by Dolev et al [18] on which it is based.

Performance of Network Management Mechanisms

- Reduction of computation load on each node by using distributed management
- Reduction of management information flow by using distributed management
- Reliability at least similar to centralized network management approach
- Self-adaptive behaviour optimizes available resources

The trade-off between accuracy and protocol overhead of the protocol can be controlled by changing the parameter λ_g, which is the rate under which nodes undergo a protocol cycle. With a low rate, fewer messages are exchanged and fewer updates on local state are done, reducing computational load, at the cost of the accuracy of counting.

Resilience to failures is one of the fundamental properties of the algorithm. Also, during intervals without churn, the root obtains the correct count in time linear in the diameter of the network graph.

Any node that becomes available begins to contribute to making the global aggregate count more accurate by performing neighbourhood sampling.

6.1.3 Threshold Crossing Detection

TG-GAP is a distributed protocol for monitoring the detection of threshold crossing by a global aggregate. Global aggregates are computed from local (device) metrics through some aggregation function (e.g., SUM, AVERAGE or MAX). The protocol raises an alert when a monitored aggregate grows above some threshold and clear the alert when the aggregate falls below a smaller hysteresis threshold. For example, TG-GAP can raise an alert when the total number of VoIP calls crossing a domain, as aggregated across IP PBXs, grows above 20,000 and clear the alert when it falls below 15,000.

Our approach for distributed detection of threshold crossing is to start with a gossip-based protocol for computing global aggregate, and extend it to support detection of threshold crossing. The protocol uses local thresholds to decide whether or not nodes gossip with their neighbours. The effect of this is to reduce the protocol overhead during periods where the monitored aggregate is far from the threshold. The underlying gossiping mechanism is also used to compute snapshots of the aggregate and to synchronize local states of the nodes.

Assumptions

- There exists a distributed management architecture, whereby each node in the networked system participates in the monitoring task by running a monitoring process, either internally or on an external, associated device.
- Local metrics can be accessed on each node, where they are periodically updated in an asynchronous fashion.

Evaluation of Requirements

Information Gathering and Collection

- Situation awareness, detection of network conditions
- Detection of network anomalies

TG-GAP continuously monitors aggregates for threshold crossing. Depending on the monitored aggregate and the value of the threshold, an alert by TG-GAP indicates to the management system a specific network condition (e.g., average link utilization above 50%) is met. This condition may be indicative of a normal network operation or the result of an anomaly.

Information Distribution and Node Collaboration (by Node Interaction)

- Distributed management and role based interaction

TG-GAP is a distributed algorithm where each node in the networked system participates in the monitoring task by running a monitoring process. More specifically, all nodes run (independently and asynchronously) the same algorithm such that the states on all nodes converges to the correct one.

Core Network Management Requirements

- Management based on situation, policies and/or business objectives

One of the design goals of TG-GAP is to avoid management traffic when the monitored aggregate is far above or below the threshold. This is an important property of a threshold detection protocol since, in cases where alerts are associated with overload situations, it is important that management does not worsen the situation through generating more overhead in the managed system.

Performance of Network Management Mechanisms

- Reduction of computation load on each node by using distributed management
- Reduction of management information flow by using distributed management
- Reliability at least similar to centralized network management approach
- Substantially faster adaptation compared to centralized approach

As can be seen in Figure 6-2 below, compared to a naïve protocol that continuously monitors the aggregate, TG-GAP consumes at most an order of magnitude lower overhead when the value of the monitored aggregate is less than 80% of the threshold. In addition, through a configuration parameter of our protocol, we can effectively control the trade-off between the quality of the threshold detection and the protocol overhead. (A lower protocol overhead means that each node has to process fewer messages and therefore the computational load on each node is reduced.)
The distributed nature of TG-GAP makes it resilient to churn. For instance, our simulations show that in a network where up to 1% of the nodes are either added to or removed from the network every minute, the protocol exhibits no discernable performance degradation, compared to the case where no failures occur.
Finally, unlike centralized systems, the evaluation of the performance of TG-GAP shows that the protocol overhead and the detection delay of threshold crossings grows sub-linearly with the system size.

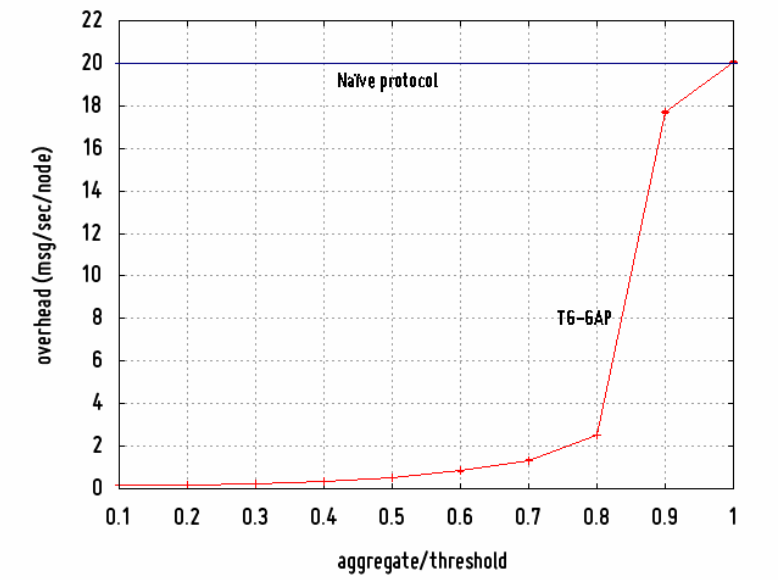


Figure 6-2: Comparison of TG-GAP with naïve protocol

We report on these and other findings in [29].

6.1.4 Private Aggregation Algorithms

Three algorithms are given for private state aggregating with application for instance for collaborative security monitoring via privacy preserving log sharing. Two algorithms are information-theoretically secure and one is computationally secure against passive adversaries ("honest-but-curious"). The algorithms have sub-linear message complexity. The functions considered are sum, max, disjunction and thresholds. The algorithms are:

- Information-theoretically secure summation and averaging. The algorithm works over incomplete networks and requires that the adversarial structure does not separate the network graph.
- Computationally secure disjunction of local inputs. The algorithm uses homomorphic encryption together with non-private summation. It returns false if all local inputs are false and gives incorrect output true with small probability. The algorithm can be used to compute maximum.
- A composition structure where a secure sum algorithm is combined with a standard algorithm for computing other functions. A computationally efficient algorithm such as algorithm 1 is used to accumulate inputs, while a more expensive algorithm is executed by a subset of nodes which jointly act as a trusted party, e.g. mutually trusted servers over multiple domains.

For details of the algorithms we refer to [20].

Assumptions

- Round-based communications and computations are assumed. A synchronous computation model is implicit.
- A connected, but possibly incomplete, undirected network graph $G = \Box P, E \Box$ is assumed, in which P are the n participating parties. Each party P_i has a private input x_i. The input is either a single integer or a boolean value.

- Passive “honest-but-curious” adversaries are assumed – that is, adversaries which may collude to learn information about honest users inputs. Active adversaries are not considered.
- Adversarial structures are defined in graph theoretic terms. The algorithms assume that the adversarial structure does not separate the graph G. The second and third algorithms require the adversarial structure to be monotone, i.e. closed under taking subsets. The third (composition) algorithm additionally requires that there exists a complete subgraph K not contained within any union of two adversary structures.
- The second (disjunction) algorithm requires a homomorphic cryptosystem, resistant under chosen plaintext attacks (IND-CPA).

Evaluation of Requirements

Information Distribution and Node Collaboration (by Node Interaction)

- Distributed management and role based interaction
- Security, establishment of trust relationships

The algorithms allow aggregation functionality to be distributed across domain/trust boundaries for the case of passive attackers. Parties can be corrupted without bounds as long as the adversary structure does not separate the network graph. An IND-CPA homomorphic cryptosystem is required for algorithm 2. Choice of algorithm otherwise left open.

Core Network Management Requirements

- Distributed Network Management architecture

The algorithms support scalable and distributed implementation of management architectures that cross administrative boundaries.

Performance of Network Management Mechanisms

- Reduction of computation load on each node by using distributed management
- Reduction of management information flow by using distributed management
- Substantially faster adaptation compared to centralized approach
- Quick switching of network wide behaviour

Algorithm overhead to support privacy is - at least in common cases such as summation and averaging - negligible, and so all the properties of the scalable, distributed solutions developed in WP4 are supported also in the case of private aggregation. Only one initial round of randomization is needed, and the algorithm can be used with any centralized or decentralized algorithm for performing the actual aggregation.

6.1.5 Adaptive Avoidance of Network Implosion

"Not All aT Once!" (NATO!) is a probability scheme and algorithms for precisely estimating the size of a group of nodes affected by the same event without explicit notification from each node, thereby avoiding feedback implosion. The main idea is that after the event takes place, every affected node waits a random amount of time taken from a predefined distribution, before sending a report message. When the gateway receives sufficient messages to estimate the number of affected nodes with good precision, it broadcasts a STOP message, notifying the nodes that have not reported yet, not to send their reports.

The gateway then analyzes the transmission time of the received reports, defines a likelihood function, and uses the Newton-Raphson method to find the number of affected nodes for which the likelihood function is maximized.

We provided algorithms for the nodes (start timer, obtain a random delay, and if not received stop, send its report) and for the gateway (receive sufficient reports, broadcast STOP, estimate the size of the group). Further information can be found at [21].

Assumptions

- The network consists of a large group of end nodes reporting to a single gateway. The number of affected nodes that should have sending their reports is large.
- Report messages are identical (e.g. an acknowledgement or a negative acknowledgement to a request)
- The gateway is able to broadcast a STOP message
- The setup allows for precise timing, i.e., (a) the event occurs at the same time, or the server can start NATO! by means of a START broadcast message; (b) all nodes are time-synchronized, or the network delays are known

Evaluation of Requirements

Information Gathering and Collection

- Monitoring of lower layer information
- Situation awareness, detection of network conditions
- Detection of network anomalies
- Characteristics of devices
- Information about network resources

The NATO! scheme can be used to estimate the number of nodes that have something in common. For Example, node that are experiencing a specific metrics beyond a threshold value (e.g. SNR ratio beyond a specified value), or are configured in a certain way, or are capable for responding to a specific request (e.g., can accommodate QoS parameters for a requested service).

Information Distribution and Node Collaboration (by Node Interaction)

- Multicasting status and capabilities of nodes

NATO! can be an alternative to multicasting. Rather than multicast a request for all nodes to respond, the gateway can send a NATO! request that is formulated with expected Y/N response only (e.g. which nodes have a specific capability, which node meet a special state). Only a small number of nodes that match the request will respond, and the gateway can accurately estimate the total number of such nodes.

Core Network Management Requirements

- Self-Adaptation of network components

NATO! can facilitate self-adaptation. In response to the current state of the network, the gateway can modify the type of queries it broadcasts, the threshold value of a parameter, or the implicit time gap in which NATO! is started. See [24], section 6.3.2, INM applications for NATO!.

Special, Data Communication Related Requirements

- Routing related functionality: Route discovery or self-rerouting

NATO! enables QoS routing, as explained in the QoS example in [24], section 6.3.2, INM applications for NATO!. The NATO! scheme can conclude the number of nodes that are capable of QoS routing, and those that can accommodate a flow with specific QoS requests. This information assists the gateway in setting appropriate routes.

Performance of Network Management Mechanisms

- Reduction of management information flow by using distributed management
- Quick switching of network wide behaviour
- Self-adaptive behaviour optimizes available resources

NATO! clearly reduces management information flow; only a small number of nodes respond to a query/state change. While the scheme utilizes a central management station, the algorithm distributes the responsibility and execution code to all participating nodes in the network domain. The Scheme enables scalability; it can be deployed in any network size, as it drastically reduces the number of messages. And lastly, NATO! supports self adaptation, as already explained above.

6.1.6 Topology Discovery

Discovery of nodes and the network topology are issues already addressed in the literature. In most of them, the nodes send broadcast messages to all neighbour nodes to obtain information from the topology of network. The major drawback of this approach concerns on overhead of messages and synchronization of local repositories. For an efficiently topology discovery process there must be suitable mechanisms that spend low cost of communication in a distributed way. We propose *Hide and Seek* (H&S), a new algorithm for network discovery, information propagation and synchronization, betting on the directionality of choices in distributed way. The roles in our algorithm are well-defined. This means that the INM (seeker) has the specific function to seek other entities, while the INM (hider) is a hidden entity at the moment. Each entity does not have a prior knowledge of their neighbours, and the communication between INM entities depends on their neighbours to relay messages on their behalf. In this process there is a high level of collaboration and cooperation between the entities. Our topology discovery algorithm can work in the bootstrapping process of each INM (Seeker) entity. Thus, H&S ensures the communication of relevant and sufficient information on each entity to ensure decision-making processes. H&S can ensure multiple collaboration groups, where each entity needs to choose which member will participate.

Assumptions

- The bootstrapping process starts with at least one INM (Seeker) entity, and their characteristics can be fixed or be changed due the upgrades.
- Network topology can be wired or wireless.
- Each entity has a well defined role in the algorithm, e.g. INM (Seekers) are responsible to discovery INM (hiders) entities.
- All gathered information is recorded in local repositories and it is periodically synchronized.
- All information flows between the INM entities uses these messages: initial message contact, response and synchronization).

Evaluation of Requirements

Information Gathering and Collection

- Characteristics of devices
- Information about network resources

The H&S has a local repository database that stores local node information (e.g., IP, MAC, mobile/fixed device, network interfaces, physical transmission type, neighbours contacted, etc.). Each repository is created locally, and the node is responsible for adding, updating and refreshing all gathered information during the discovery process. The *Repository Information Control* (RIC) controls the repository and is used to classify the type of information, e.g. available resources, network size-awareness, network domain diameters, network device type (mobile or fixed) etc. The RIC function guarantees integrity and readiness of information of each INM entity and ensures that only relevant information is recorded in each INM entity repository. On the other hand, the algorithm begins the *Mapper Nodes and Resources Discovery* (MNRD) function, obtaining specific information of each available resource in the INM (seeker). After this process, the new information is synchronized on both INM (seeker) repositories. In addition, each INM (seeker) node has an internal identifier that performs entity differentiation into the network.

Information Distribution and Node Collaboration (by Node Interaction)

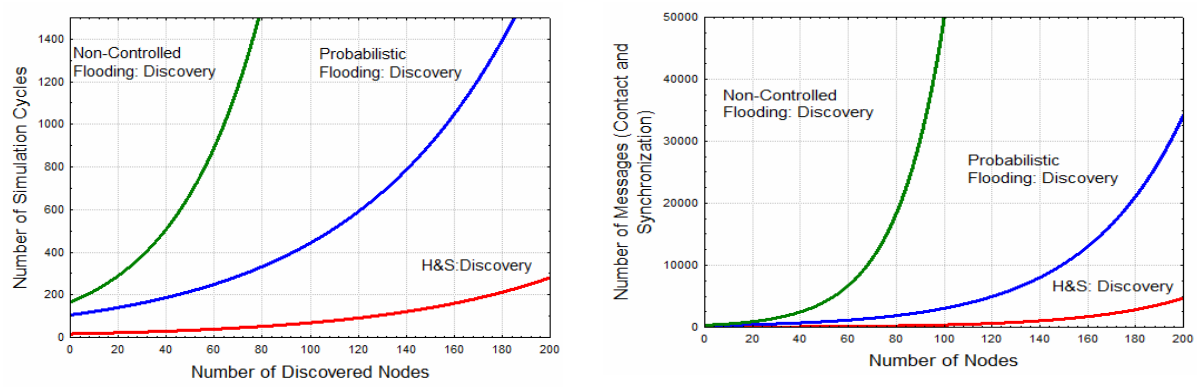
- Distribution of captured and collected information
- Information exchange done in a standardized way
- Distributed management and role based interaction

As already mentioned above, the roles in H&S are well defined, and each INM (seeker) entity receives an answer to the contact message and sent to an INM (hider) entity. After receiving the answer, the INM (seeker) entity gets INM (hider)'s local information, synchronizes the information on the repository, and changes the status of INM (hider) to the new INM (seeker) into the network. The *Probabilistic Eyesight Direction* (PED) chooses the optimal direction of the search based on neighbour's information of initial starting point (e.g, first node INM (seeker) to begin the search). We assume a starting point node Sp that has k neighbour entities (e.g., k means the amount of entities that are in the surrounding area of Sp). This step is complete when all INM (hider) entities contacted become new INM (seeker) entities. H&S ensures strong cooperation between INM (seeker) entities due the periodically signalling of messages between the entities. If a new knowledge is gathered, only this new information is recorded. Hashing techniques are used to ensure that only the new information is recorded into the local repository of each INM (seeker) entity.

Performance of Network Management Mechanisms

- Reduction of management information flow by using distributed management

As shown in Figure 6-3 H&S can reduce the amount of simulation cycles and overhead of contact and synchronization of messages through the PED function [26]. The H&S can be adaptive in any network size and it works well when the network is huge in size. This factor is proved due the linearity nature of algorithm.



(a) Amount of simulation cycles (b) Overhead of contact and synchronization messages

Figure 6-3: Performance of H&S algorithm based on simulation cycles and message overhead

6.1.7 Search in Dynamic and Self-organizing Networks

Search methods in dynamic networks usually cannot rely on a stable topology from which shortest or otherwise optimized paths through the network are derived.

Exploration and search methods are required to enable services and content retrieval in communication networks on all layers and for various purposes. Even in fixed network areas of the Internet, where the topology is stable enough to establish standard routing protocols and search engines to locate nodes and information on them, they have to cope with continuous changes. Much more dynamic network structures are often encountered on peer-to-peer (P2P) overlays as well as in wireless sensor or mobile ad hoc networks (MANETs).

When dynamics is high enough to disable reliable search indices and routing tables, other methods like flooding or random walks have to be considered to explore the network. Gossiping methods as proposed for monitoring in the INM framework can also be used. Random walks can exploit partially available information on network paths, but the search effort naturally increases with the lack of precise path information due to network dynamics. This problem is especially relevant for wireless technology with strict limitation on power consumption.

We compare the efficiency of random walks and flooding for exploring networks of small to medium size. Several scenarios are considered including partial path information support for search. Transient analysis and a stochastic bound are applied in order to evaluate the messaging overhead.

Assumptions

- A search may refer to users, network nodes, information, content or services of any kind residing on network resources based on identifiers like IP addresses or hash values used in P2P networks. Although a single node is addressed as the base case, the approach is also extended to a target node set, where each node in the target set is able to respond. Other cases, where several nodes have to be involved to get a result in a production chain or a distributed scheme, are for further study.
- We focus on dynamic networks with a planar graph structure, such as sensor and mobile networks, where simple random walks are often less efficient than flooding.
- Basically, flooding spreads a request from a node to all its neighbours which repeat to it in order to contribute into an exhaustive flood covering the entire network. When the same request is received several times at a node from different neighbours, then only the first receipt is forwarded and later ones are discarded. Wireless networks spread messages through broadcasting over a limited range. Therefore only one message is

required per node for flooding, whereas several messages are sent in overlays or meshed point-to-point networks.

- Basic random walks choose the next hop in a network with the same probability among all reachable neighbours. The transient analysis used to evaluate the performance of random walks also can include partial information that is assumed to be available on the nodes. In this case of biased random walks, the progress may be partly deterministic and only partly randomized. Random walks often can reduce the communication overhead, but they traverse the hops sequentially and thus usually spend much more time than flooding. Multiple random walks in parallel are included as a compromise between demands for low delay and low overhead.
- The model does not cover all realistic scenarios, since it is left open how a node decides if it has valid up-to-date information to direct the search to the next hop towards the target. In addition, a homogeneous information distribution is implied, whereas networks structures are often inhomogeneous, e.g. hierarchical, and information is usually more precise near the target.

Evaluation of Requirements

Information Gathering and Collection

- Monitoring of lower layer information
- Situation awareness, detection of network conditions
- Information about network resources

The methods in use depend on INM information gathering and collection methods employed on lower layers. Whenever a path towards target nodes is uniquely identified by information on the network entities, more expensive exploration steps can be saved.

Information Distribution and Node Collaboration (by Node Interaction)

- Common Information model and protocols
- Distribution of captured and collected information
- Information exchange done in a standardized way
- Distributed management and role based interaction

The study is focused on environments where information distribution and node collaboration is challenging because of node mobility, churn and other dynamic processes detracting from stable and reliable network structures.

Core Network Management Requirements

- Distributed Network Management architecture
- Make decisions and take actions

Currently the IEFT standardization has a highly active working group on routing over low power and lossy networks (ROLL WG). When the ROLL WG succeeds to establish new routing methods for this environment, then decentralized network management approaches will be set up on top of them as a next step of Future Internet standardization.

Special, Data Communication Related Requirements

- Routing related functionality: Route discovery or self-rerouting

The search for information usually does not have much impact on traffic volume, which is more driven by a download or data distribution phase that may follow the search phase. Message exchange for search can benefit from preference given in a differentiated service scheme as is also usual for routing or signalling traffic.

Performance of Network Management Mechanisms

- Reduction of computation load on each node by using distributed management
- Reduction of management information flow by using distributed management
- Reliability at least similar to centralized network management approach
- Substantially faster adaptation compared to centralized approach
- Self-adaptive behaviour optimizes available resources

In highly dynamic networks, distributed management often is the only method of choice rather than just an enabler for performance improvements. When nodes have moved away randomly or disconnected while information about them is still on the way to a remote management centre, then centralized management is neither reliable nor useful at all. In addition, the paths from transient nodes to a management centre are steadily changing or temporarily interrupted due to churn in the availability of other intermediate nodes. Therefore “reliability at least similar to centralized network management” is easily obtained but does not present a reasonable criterion.

6.1.8 Anomaly Detection

Distributed anomaly detection is based on a probabilistic approach to self-adaptive monitoring and detection of network variations. Essentially, a statistical method is used to model the local network behaviour that is monitored in a fully distributed manner. This approach allows for increased adaptivity to long-term network development as well as reduced requirements on manual configuration. In particular, the method is developed for detecting symptoms of faults and shifts in behavioural patterns, independently of any specific network condition, by autonomously adjusting detection parameters in local regions of the network. In general, autonomous and adaptive methods for fault-handling and anomaly detection are important elements for maintaining reliable and failure-resilient networks. The self-adaptive properties of the algorithm are here related to configuration of probing- and detection-mechanisms. Probing is used for observing and modelling various aspects of the local network situation, e.g., expected latency and packet loss on each connection. Adaptation to network variations is achieved using overlapping statistical models such that old data are successively replaced with new data. Based on the current network conditions and high-level management requirements, the algorithm behaviour is adapted (similar to e.g., A-GAP). Detected faults are isolated to a certain link or node via collaborative fault localisation between nodes. Results obtained from performance tests of the algorithm are reported in D4.3.

Assumptions

- It is assumed that there are means to perform measurements on different aspects of the network, e.g., network delays and packet loss.
- It is assumed that each node can obtain topological information, to be aware of all neighbouring nodes within the topological distance of two (e.g., with input from topology discovery).

Evaluation of Requirements

Information Gathering and Collection

- Monitoring of lower layer information
- Characteristics of devices
- Situation awareness, detection of network conditions
- Detection of network anomalies
- Information about network resources

The method is developed to detect anomalies in various aspects of network communication, based on end-to-end measurements between nodes. This can be done on any levels without requiring rigorous modifications of the algorithm. The local measurements are statistically modelled individually for each connection, and adapts to the local network behaviour and the properties of the network equipment, as it develops over time. Based on the statistical models, faults, anomalies and changes in the network can be detected, to maintain situation awareness. Finally, anomaly detection supports information about network resources, such as availability status and statistics.

Information Distribution and Node Collaboration (by Node Interaction)

- Distribution of captured and collected information
- Distributed management and role based interaction

The algorithm detects anomalies in a collaborative manner between nodes. Availability status of a certain node or link is shared between other nodes in the immediate neighbourhood. Anomaly detection supports distribution of collected statistics, in the sense that anyone interested in such information can request it. Moreover, detected faults are pinpointed to specific network equipment. This allows for preventive actions in order to maintain quality of service, as well as efficient fault-handling and management.

Core Network Management Requirements

- Management based on situation, policies and/or business objectives
- Make decisions and take actions
- Self-Management e.g. according to FCAPS model
- Self-Adaptation of network components

The approach supports situation based management and decision making, as it provides availability status and information about observed behaviour in different regions of the network. High-level parameters expressing detection certainty and probing costs relative estimations on individual links support policies and business objectives. In addition, guarantees on maximum and minimum detection delays and probes needed to confirm a fault can be enforced if necessary. Further, the algorithm supports fault-management in terms of detection and localisation of abnormal behaviours, and is also self-adapting over time, in the sense that low-level algorithm parameters are self-configuring. The information provided by the algorithm, supports self-adapting behaviour in network components.

Special, Data Communication Related Requirements

- Traffic differentiation and handling
- Routing related functionality: Route discovery or self-rerouting

Anomaly detection supports traffic differentiation and handling, as well as dynamic routing and route discovery, as it reports availability status for individual network components.

Performance of Network Management Mechanisms

- Self-adaptive behaviour optimizes available resources
- Quick switching of network wide behaviour

The algorithm can autonomously configure its low-level monitoring parameters (e.g., probing intervals and number of probes needed for fault detection) to locally observed network behaviour, which can increase communication efficiency [15] [24]. Due to its distributed nature, the algorithm can be deployed in most types of networks of any size (traffic scales linearly with the number of connections [24]) with only minor modifications. The approach can in an adaptive manner successfully detect abnormal behaviour and localise it to certain network components [15][24]. For example, simulation results show that over 95 % of generated communication faults can be detected, whereas the detection performance is around 90% of the generated latency shifts. Further, a localisation performance of up to 80-90% was achieved in performed experiments. The information provided by the algorithm therefore supports resource optimization and adjustments of network behaviour in different regions of the network.

6.1.9 Aggregation for Reputation Systems

D-CAF (Distributed Context-Aware Firewall) is able to react to overload situations and generate automatic firewall rules. Based on the network's workload and subjective user analysis and valuation (taken as context information), it is capable of taking fast filtering decisions in order to keep protected services available for the most trusted users, even under overload situations generated by malicious attacks (such as DDoS).

The decision-making module (which may be installed on the gateway router, as well as somewhere else as a standalone node) gathers traffic conditions information while, at the same time, protected services will send subjective user valuations based on policies and/or business objectives to this module. In case of an overload, the module will generate just the necessary amount of filters in order to keep the protected link's traffic under a defined threshold. Filters are generated using the valuation information, going up from the worst valuated users until the overload situation is under control.

This approach delivers a quick-response measure for critical overload scenarios, in a way that administrative personnel hasn't been able to apply until now.

Assumptions

- Protected Services are able to analyze and valuate incoming traffic. (Policies are to be defined by their administrators)
- User's valuations describe a cost/benefit ratio based on business or administrative interests. Therefore, well valuated users are valuable users.
- Traffic under threshold = no reaction. Attack or not, it is not of this system's particular interest to block malicious users. At contrary, it aims to remain available for trusted and/or valuable users.

Evaluation of Requirements

Information Gathering and Collection

- Situation awareness, detection of network conditions

- Information about network resources

As described, the system gathers information about network congestion and cost/value ratio of users.

Information Distribution and Node Collaboration (by Node Interaction)

- Common Information model and protocols
- Distribution of captured and collected information
- Information exchange done in a standardized way
- Security Framework, establishment of trust relationships

The process of gathering user valuations results in a valuation matrix, which can be transparently exchanged between many instances of D-CAF across the network.

All communications between components, namely decision-modules, traffic measurement probes (if applicable) and the protected services are made using the IPFIX protocol. Every component may be identified to each other and even between separate, mutually trusting networks, in order to securely exchange measurement and/or valuation information. This way, every instance of D-Caf works as a building block for a wide-range overload protection framework.

Core Network Management Requirements

- Management based on situation, policies and/or business objectives
- Self-Adaptation of network components
- Make decisions and take actions

Valuation policies have to reflect policies and/or business objectives, that way the filtered users will be, at least in their majority, the least interesting ones to keep online during an overload situation.

Special, Data Communication Related Requirements

- Traffic differentiation and handling

The protected services valuate their users, resulting in a valuation matrix which enables the system to differentiate and control traffic.

Performance of Network Management Mechanisms

- Self-adaptive behaviour optimizes available resources

Actions taken by this systems aim at resource availability during legitimate or malicious overload scenarios. During this period of time, filtering some users enables services to remain available at least for the “better” users, instead of being unavailable for all of them.

Figure 6-4 shows the relationship between the expected amount of filters for wanted and unwanted traffic. As one can see, the number of filtered legal addresses grows insignificantly in comparison with the filtered bot traffic. For more information, please read [42].

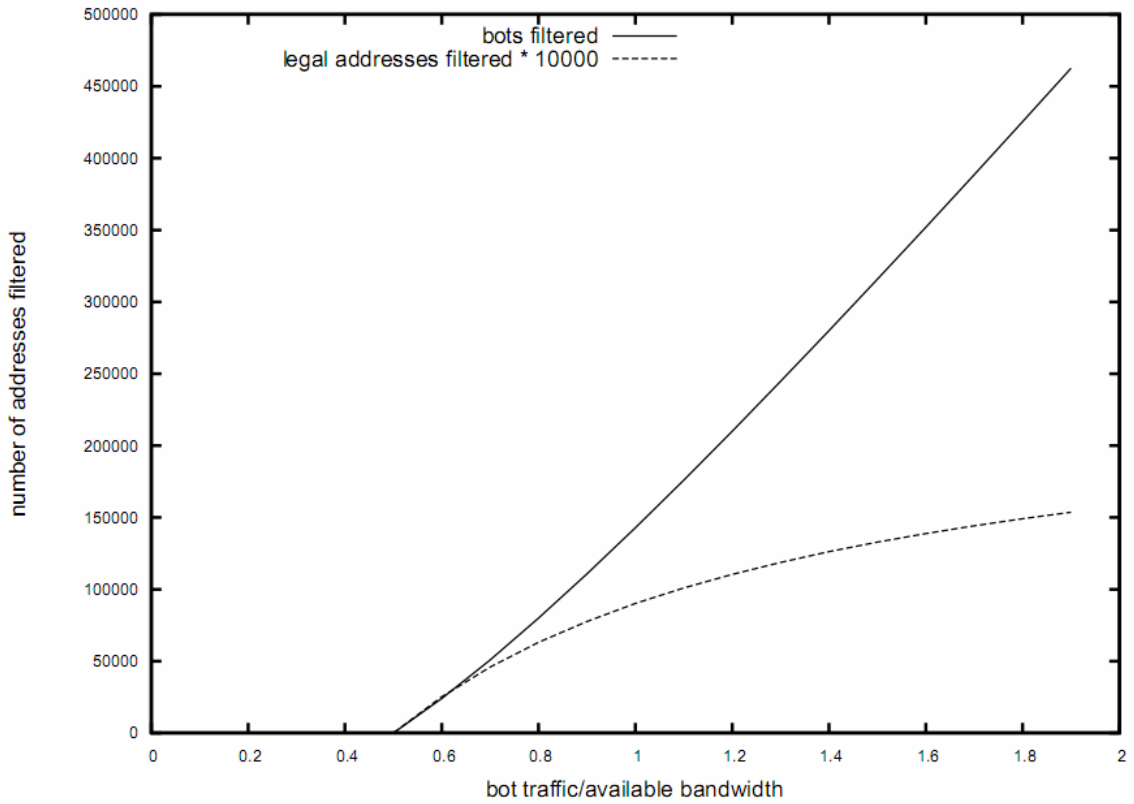


Figure 6-4: Filtering curves for wanted and unwanted traffic

6.1.10 Wireless Network Monitoring Supporting Routing

The well known approaches to network monitoring have limited application to wireless mesh networks (WMN). The main problem lies in complex behaviour of underlying mechanisms regarding the physical layer as a result of mutual flows interferences and so called flow self-interferences (i.e. the hidden and exposed terminal problem). The classical monitoring approach in which the path parameters (the path routing metric) can be calculated on a per link basis (addition, multiplication etc.) leads to improper results. In order to cope with the problems mentioned above we proposed a new approach in which the paths monitoring is based on cross-layer information, it is separated from routing and is a part of INM. It is worth mentioning that in the proposed approach the path quality is not only monitored, but also predicted. It provides the ability of the dynamic switching of the routing metric and simultaneous use of multiple routing protocols. The more detailed description of the proposed approach, references and some simulation results are presented in Deliverable D4.3. It is easily integrated with other INM monitoring mechanisms, and according to the presented idea, it can be easily upgraded without modification of other components of the architecture.

Assumptions

- It is assumed that the multi-hop wireless network is based on nodes equipped with 802.11 radio interfaces (single or multiple)
- It is assumed that the proactive routing is used and the path metrics are part of the routing table that is accessible to INM
- The multi-path proactive routing protocol which adapts forwarding of the network state is preferred whereas the proposed approach is not limited to multiple paths
- The network may use single or multiple routing protocols simultaneously

- The routing protocols are able to use multiple parameters of the routing metric (i.e delay, SNR, Hop Count).

Evaluation of Requirements

Information Gathering and Collection

- Monitoring of lower layer information
- Situation awareness, detection of network conditions
- Information about network resources

As described, the system gathers information about paths quality, and is independent on the routing protocols. The information about the network state is obtained via monitoring and prediction. The measurements are done on a per link basis. The proposed approach is tightly coupled with the entity being managed because it monitors the physical layer (the inherent level of embedding). The monitoring algorithm parameters (monitoring intervals, accuracy, etc.) can be controlled according to network usage levels, policies and/or business objectives.

Information Distribution and Node Collaboration (by Node Interaction)

- Distribution of captured and collected information

The proposed method is designed to provide the paths monitoring independently on the routing protocol. It takes into account the lower layer information. The monitoring information is calculated on a per link basis and disseminated after appropriate processing along the routing paths – there is an aggregation/processing of the monitoring information during dissemination. Due to it distributed nature, the algorithm can be deployed in wireless mesh network of any size (no scalability problems) with minimum modifications.

Special, Data Communication Related Requirements

- Routing related functionality: Route discovery or self-rerouting

The routing protocols should be able to read the path quality estimation data proposed by the new approach. There is only one requirement regarding the routing protocols cooperation with INM in this context. The algorithm is a part of INM and is interfaced with routing protocols via the routing table and can be used for self-rerouting in load adaptive routing. The information about the paths state can be also used for rerouting combined with fault discovery.

Performance of Network Management Mechanisms

- Self-adaptive behaviour optimizes available resources

Due to the usage of the proposed approach combined with adaptive/multi-path routing the probability of congestions in the network is minimized and the usage of the existing resources is optimized – i.e. for traffic forwarding and longer paths with lower load can be used. Due to the cross-layer approach the radio resources are measured more accurately. The proper behaviour of the algorithm has been confirmed by simulations described in D4.3.

6.2 Evaluation of Self-Adaptation Algorithms

This section evaluates the following self-adaptation algorithms:

Benchmarking of distributed schemes, section 6.2.1, studied to which extent distributed processing is beneficial for network management algorithms.

Decentralized probabilistic management algorithms that address some shortcoming aspects of a centralized management system are evaluated in section 6.2.2.

An event handling paradigm for configuration-less dissemination of information through events in a distributed environment is evaluated in section 6.2.3.

INM processes that use chemical network protocol design are evaluated in section 6.2.4, in which molecule-like entities demonstrate execution flows that are analyzed as if they were chemical processes.

An algorithm to ensure INM stability that implement embedded verification of configuration changes is evaluated in section 6.2.5.

Cross–layer self-adaptive QoS-aware routing algorithms for VNets are evaluated in section 6.2.6.

An emergent-behaviour-based congestion control algorithm is evaluated in section 6.2.7, utilizing a concept borrowed from pulse-coupled oscillators that emergently synchronize themselves.

Finally, Self-adaptive routing algorithms for wireless multi-hop networks are evaluated in section 6.2.8.

6.2.1 Benchmarking of Distributed Schemes

This research was conducted during the first year of the project, and reported in Deliverable D4.2. It studied to which extent distributed processing is beneficial for network management algorithms. Examining a number of case studies, the research looks for the optimal point of INM distributed processing, considering the network environment and performance tradeoffs of scalability, robustness, adaptivity and overhead.

Legacy network management systems are primarily centralized, and are controlled by humans. Consequently, such systems are not scalable and are not exploiting any opportunities for automation. Our research studies the costs and benefits of distributed self-management, as compared with centralized systems.

We studied three network test cases:

- Route protection for reliable networks (scenario 2 of deliverable D4.1). Route protection is a reliable mechanism, which reserves prearranged backup paths to accommodate fast link restoration with QoS. In the event of link failure, the backup link is immediately activated, with minimal service disruption. Such schemes are traditionally adopted in MPLS-based routing domains and are mainly used for multimedia applications, where IP-based routing recovery mechanisms are too slow to react.
  We studied 6 different protection schemes, ranging from fully centralized to fully distributed, and evaluated them with regards to throughput, path restoration time, overhead and robustness.
- Adaptive data collection in sensor networks (scenario 1 of D4.1). A sensor network has a limited amount of data collection resources. Depending on the specific situation which varies over time, there are several levels of interesting measurements (referred

to as threats). It is desired that sensing resources are assigned dynamically in an optimal manner, in order to provide the best information for the specific situation. We defined a formal model for the problem, and a number of algorithms. We compared the centralized and the distributed algorithms, considering coverage, adaptivity, robustness, scalability and overhead.

- Topology discovery (scenario 1 and 2 of D4.1). A key part of real-time monitoring is topology learning, which must be very efficient, in order to provide real-time topology information, with minimal/affordable overhead. We looked at a sensor network and a large switched Ethernet network. We employed centralized and distributed algorithms, and compared them, with regards to efficiency, speed, and robustness.

Taking the results of all three test cases, we compiled a list of guidelines for a clean slate approach.

Assumptions

We believe that there is no general benchmark for all network scenarios. For each network management task and network structure, a different level of distributed effort is favoured. Taking a bottom-up approach, we are testing a few network management test cases, and try to locate the "sweet spot"; that is, the optimal point for the extent of distributed effort, considering cost-performance tradeoffs. Both qualitative and quantitative metrics are sought.

The 4WARD project exploits both clean slate design and self management. Since it is difficult to evaluate self management in a clean slate environment, our research adopted the following two-steps approach. In the first step we studied distributed self management for a few existing network scenarios, and for each one, distributed processing benchmarks were established. In the second step, we derived guidelines for distributed self-management under a clean slate solution.

Evaluation of Requirements

Information Distribution and Node Collaboration (by Node Interaction)

- Common Information model and protocols
- Distribution of captured and collected information
- Information exchange done in a standardized way
- Distributed management and role based interaction

Our benchmarking research studied to which extent distributed INM processing is beneficial. Node collaboration is a key part of distributed processing, which employs common protocols and standard means, for distribution of information among collaborating nodes.
As an example, we show in Figure 6-5 numerical results for the adaptive data collection test case that were previously reported in 4WARD deliverable [23]. The figure shows the coverage of a few centralized and distributed algorithms, as a function of total number of targets (monitored items).

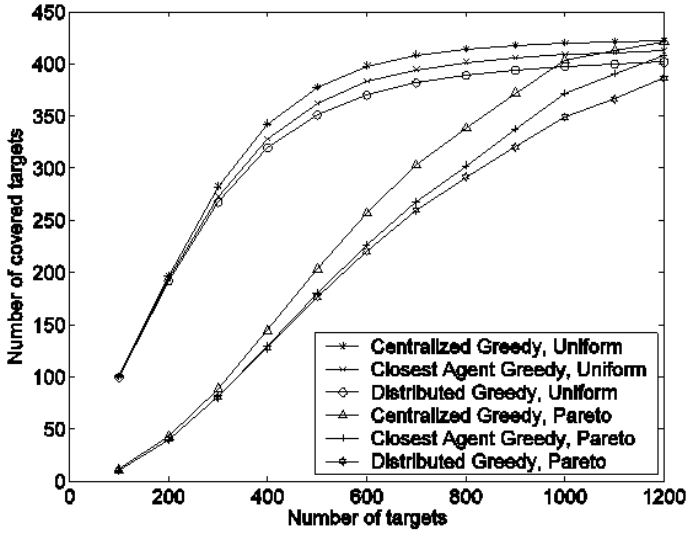


Figure 6-5: Coverage evaluation of the adaptive data collection test case

Table 6-2 shows the results of this analysis, comparing the centralized and the distributed implementation in a few categories. Using the results accumulated from all three test cases, we compiled and reported in [23] a set of recommendations for a clean slate approach, namely when and to what extent distributed implementation should be used.

Method	Centralized	Distributed
Coverage	Higher	Lower (but comparable)
Adaptivity	Slow	Fast
Robustness	Low	High
Scalability	Low	High
Overhead	Low	High

Table 6-2: Results of adaptive data collection test case

Core Network Management Requirements

- Management based on situation, policies and/or business objectives
- Self-Management e.g. according to FCAPS model
- Self-Adaptation of network components
- Distributed Network Management architecture
- Make decisions and take actions

Our research addresses network management procedures, which are self-managed, distributed, situation/state aware, and self-adaptive.

Performance of Network Management Mechanisms

- Reduction of computation load on each node by using distributed management
- Reduction of management information flow by using distributed management
- Reliability at least similar to centralized network management approach
- Substantially faster adaptation compared to centralized approach

- Quick switching of network wide behaviour
- Self-adaptive behaviour optimizes available resources

In Deliverable D4.2 [23], our study provided guidelines to which extent distributed network management processing is beneficial. The criteria used include overhead (computation load, information flow), reliability (robustness), speed of adaptation, the scope of the management activity (local or network-wide), and optimization of resources.

6.2.2 Decentralized Probabilistic Management

Decentralized management systems operate on more or less dynamic networking environments. Basically, dynamic behaviour does not allow for an exact real-time view of the system and coordination between management functions. For example, many coordinated decentralized algorithm fail in dynamic environments such as routing protocols or peer-to-peer system, but uncoordinated systems might perform better in those environments. INM has proposed a probabilistic approach for distributed network management, where a certain degree of uncertainty exists in the enforcement of certain management functions on different nodes.

A major implication in the aforementioned approach is the extent to which the overall behaviour of the network is affected: in fact probabilistic management must achieve at least as good results as with conventional network management.

Decentralized management functions typically accumulate network management information from the network, subsequently storing and computing the information for analyzing the history, to finally deriving conclusions for taking some actions. The management functions also coordinate with the same function on other nodes, or with other functions on the same or in collaboration with other nodes. The storage, computation for analysis, and communication requires some resources on the network and the nodes. If all the nodes run those functions, all of them store information and compute on them, potentially wasting quite a bit of CPU, memory, and communication bandwidth.

Assumptions

We assume that self-adaptation is performed in a distributed manner and a set of different management functions are running on each node simultaneously. Each of these functions is controlled by a randomization process, which randomly turns on or off certain management functions on the node. This is a prerequisite for resource efficient decentralized network management, because it tries to prevent redundancy in gathering and processing network management information. It allows for an uncoordinated way of achieving similar management effects as with a coordinate approach to decentralized network management.

We applied this approach to the NetInf architecture, where applications publish information and are able to get management information out of the system.

Evaluation

Core Network Management Requirements

- Self-Management e.g. according to FCAPS model
- Distributed Network Management architecture

The probabilistic approach introduces self-management in that distributed functions are coordinated independently and concurrent modifications enforced during self-adaptation can be avoided. In the preliminary study in [23], we studied the effects of probabilistic management in a well-balanced setting in order to get a feeling about the effects of the

approach in fairly normal situations. When extrapolating the monitored values to the overall network and network monitoring time, we had the same average number of information requests per node independent of the probability of running the monitoring function.

As long as the system is well balanced, we can run the probabilistic management system on very low probabilities and get an error no larger than 0.2%, i.e. the management information is quite accurate. Even when reducing the probability down to as low as 30%, which in turn means a reduction of monitoring traffic by 70%, no accuracy is sacrificed for average values across the network.

Here, we consider an unbalanced scenario, where we modified the load model to have ten dedicated nodes in the network that do a lot of information publishing. In average, they do the same amount of publishing as other nodes do retrievals, but far less retrievals than the other nodes.

Figure 6-6 shows the comparison of the standard deviations in the case of balanced and unbalanced load. In an unbalanced setting, the deviation increases faster and reaches a larger value than in a balanced setting. Hence, the monitoring accuracy is smaller. Still the average number of retrievals and the average number of publishes, when extrapolated, are the same, no matter what probability we have chosen. Although in the unbalanced case, the standard deviation grows significantly, it still remains below four percent at a probability of 0.3. This is a remarkably low standard deviation, which means that even in unbalanced scenarios, probability-based methods allow significant resource savings while retaining a high level of accuracy of the monitored information.

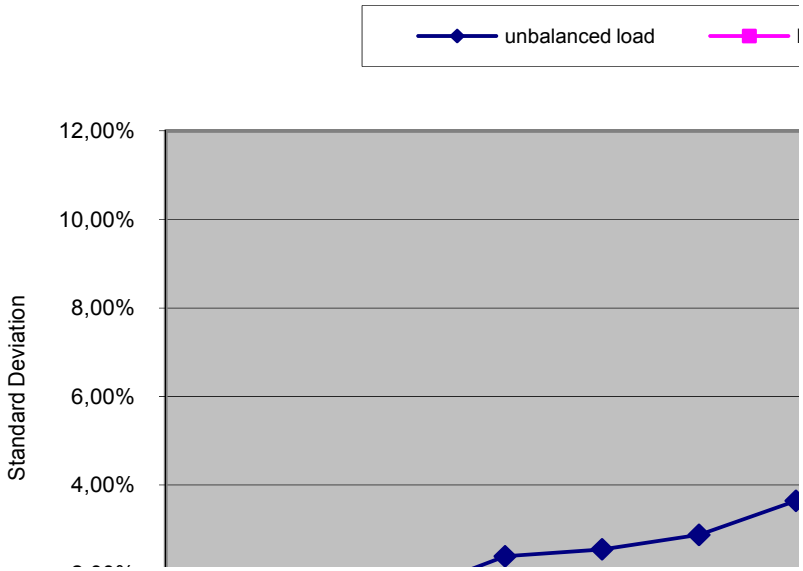


Figure 6-6: Standard deviation of balanced vs. unbalanced load.

When evaluating robustness for probabilistic management in distributed architecture, we can draw the following conclusions. In more unbalanced scenarios, the extrapolation of values is less accurate than in balanced ones, and naturally accuracy of the observed management data decreases with smaller probability values. In the presented case of fault management, we are still able to achieve a 95% success rate in the detection of faults. This value is combined from both probabilistic management and from the effect that no service requests at all occur at a subset of the faulty nodes.

The difference between balanced and unbalanced scenarios is smaller for fault management than for the monitoring case. Since fault management may be more critical in some scenarios

than just monitoring, our results demonstrate that even more critical management tasks are suitable for being subjected to probabilistic management.

Performance of Network Management Mechanisms

- Reduction of computation load on each node by using distributed management
- Reduction of management information flow by using distributed management
- Reliability at least similar to centralized network management approach

Looking at exploitations scenarios, an important aspect to consider is whether operators would accepts the concept of probabilistic management. In general, probabilistic systems lack acceptability due to their nature of being not intuitively understandable. Specifically, looking into control and management systems, there is a fairly large resistance to systems that incorporate a certain degree of randomness. If we assume that the decentralized self-managing design of future network management will become reality, we will already have a certain degree of uncertainty in the self-managing system itself, doing configuration and performance management automatically. We feel that in such situations a certain degree of randomness does not harm.

More interestingly, many of today's successful networking technologies do depend on significant degrees of randomness. One of the most prominent examples is Ethernet's highly successful CSMA/CD scheme, which employs randomization for solving distributed medium access problems. Another example is statistical multiplexing performed in IP networks. Possibly the most striking fact is that even in network security, randomization is a widely accepted and applied technique that lies at the basis of many cryptographic protocols. All of these scenarios share that the underlying systems contain certain degrees of nondeterministic behaviour, which does not allow the application of purely deterministic mechanisms. While these randomization processes are sometimes difficult to grasp, we believe that, based on our evaluation, they will also be vital in future communication networks, which will be characterized by significant complexity where indeterminacy and unpredictability will play a major role.

On the business side, it is more relevant to assess the commercial benefit of lowering the resource usage for network management and paying with less precision in some cases. It always makes sense to design resource-efficient systems, specifically, in mobile and resource-limited environments. In many novel network architectures, the management of very dynamic networks like car-to-car communication, ad-hoc networks and peer-to-peer is required. In such environments, probabilistic management helps by easing the introduction of management into the dynamic system, which would not be possible with traditional management paradigms.

6.2.3 Event Handling

An important function to support quick self-adaptation is a robust mechanism to identify anomalous conditions in the network and distribute these events in time. When applying a distributed subscription/notification mechanism to management of large scale networks, it is important to guarantee scalability and robustness.

Scalability refers to the traffic generated during the notification: when an anomaly occurs, different distributed functions might need to be discovered and notified, but the total amount of traffic must be limited to avoid congestion. Robustness refers instead to the final success of the event notification mechanism: anomalies must be correctly reported to the destination functions or, if this cannot be guaranteed, at least the overall robustness of the network shall not be impacted. In the section we report our evaluation in terms of traffic generated by our

event distribution scheme as well as delivery success rate in case of stringent time requirements.

Assumptions

We assume that the underlying event distribution mechanism is structured as described in [23]: it relies on a topology structure and on an estimation of the delays to process events. Here it is assumed that these delays are provided at boot-strap during an initial learning phase.

Another assumption is that root cause correlation is performed in a distributed manner in the network, and that events must be delivered to the appropriate analyzer in time. Additionally, the limitation in computational capabilities of a handler is considered as the number of events that it can process concurrently.

Evaluation

Information Gathering and Collection

- Situation awareness, detection of network conditions
- Detection of network anomalies

The event handling defined in INM allows for dissemination of alarms in the network. When anomalies are detected as described in section 6.1.8, this mechanism allows communication of the anomaly to self-adaptation functions in the network. The implications in terms of traffic and timeliness are part of the evaluation of the quantitative requirements reported below.

Core Network Management Requirements

- Self-Management e.g. according to FCAPS model
- Distributed Network Management architecture

As adaptation mechanism, we investigated the possibility to enable a concurrent dissemination of events, so that different root cause analyzers are reached at the same time.

Table 6-3 shows a comparison of the traffic generated in the different approaches (sequential or parallel), with decreasing computational capability. When the computational capability gets very low traffic decreases in the parallel approach, because saturated handlers do not further forward events, which reduces the overall number of messages sent. However, the consequence of this saturation is that the success rate of the system gets lower and this affects in particular the results in the parallel case. Table II shows the numeric values corresponding to the lowest computational capabilities and highlights that the percentage of successfully handled events gets lower. The result also indicates that a higher degree of parallelisation is more favourable compared to lower degrees of parallelisation or sequential approaches.

computational capability (# events)	success rate (%)		
	sequential	parallel 3	parallel 6
6	0.928	0.957	0.968
5	0.927	0.952	0.957
4	0.910	0.930	0.938
3	0.892	0.912	0.924

Table 6-3: Percentage of successfully handled events in relation to computational capability, homogeneous scenario

Performance of Network Management Mechanisms

- Reduction of computation load on each node by using distributed management
- Reduction of management information flow by using distributed management
- Reliability at least similar to centralized network management approach

When looking at the numeric results about completion time and traffic consumption, it is clear that a trade-off exists between generated traffic and success rate. The completion time for parallel approaches is much lower compared to sequential approaches. This is because sending the event to multiple handlers increases the probability that one of them is able to handle it with success. Increasing the number of destinations can further reduce the completion time, especially if the probability of success of the closest handlers is low. However, the parallel approach has the evident disadvantage that the traffic generated increases linearly with the number of parallel destinations.

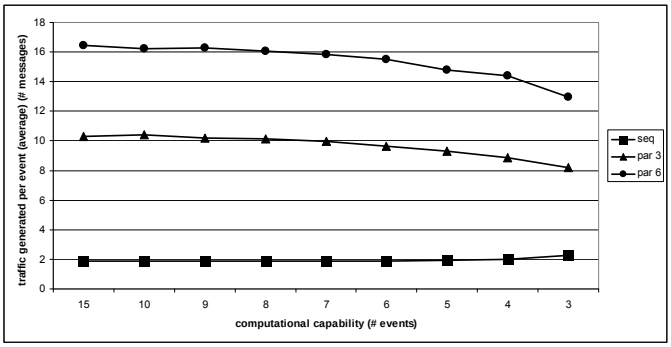


Figure 6-7 Traffic generated in relation to computational capability, homogeneous scenario

The trade-off between success rate and generated traffic cannot be resolved, but necessarily the expected behaviour of the system should be defined in the governance process of the INM framework. In addition, we propose that nodes should support the different distribution mechanisms, and they should be able to switch from one to another on the basis of network conditions. For example, during plug-and-play phase every node can be bootstrapped with the most conservative mode (i.e. the sequential mode), but then the nodes running time-sensitive services should be able to switch mode to maintain timeliness of their services.

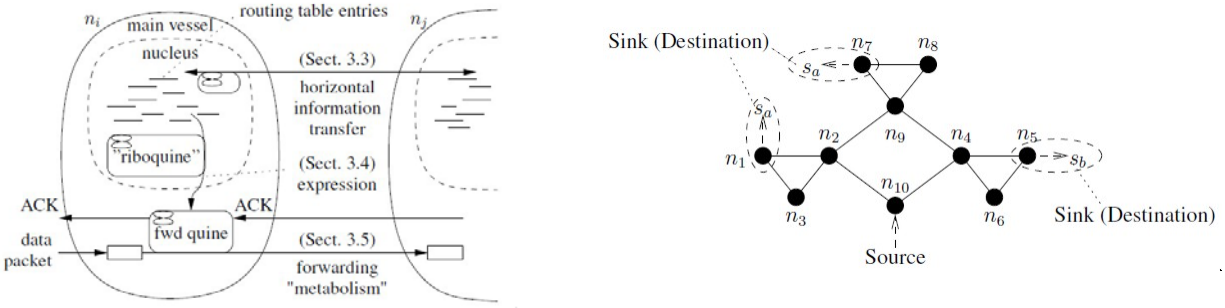
6.2.4 A Service-based Chemical Routing Protocol

Our protocol founded on chemical networking design principles and the concept of Quines, provides a routing protocol facility. It was inspired by the structure of an eukaryotic cell [36] which features a nucleus. Every network node has two reaction vessels: The main vessel features the forwarding engine for exchanging data packets with neighbouring nodes (see Figure 6-8(a). A second vessel called “nucleus” contains the “genome”, which in our case involves information about the topology of the network in form of routing table entries. Linking both vessels, there are “riboquines” (inspired by ribosomes in cells) which are responsible for “expressing” the nucleus’ routing table entries into forwarding rules in the main vessel.

Initially network nodes gather topological information about which service can be reached over which neighbour node(s). Unlike in traditional routing protocols, we do not aim at immediately finding the best path to a service but instead, the transmission paths are later reinforced by the forwarding engine. Path reinforcement is based on a competition and reward mechanism

among forwarding rules. Forwarding rules are <*destination service, next hop*> tuples represented by molecules that are likely to react with passive molecules (data packets) destined to that service. On each node one of the forwarding rules, matching the packet's destination, is randomly selected based on a probability corresponding to their relative concentrations. The forwarding rule reacts with the data packet resulting in its transmission to the corresponding next hop. When a packet finally reaches the destination service, an acknowledgement is sent back along the reverse path. This acknowledgement packet reacts with all corresponding rules that previously forwarded the data packet and triggers their replication.

In this way whenever a forwarding rule contributed to the successful delivery of a packet, it is duplicated thus increasing its concentration and also the probability of being chosen in subsequent forwarding operations. This reward mechanism results in the most efficient path to get reinforced, while less efficient paths quickly vanish due to the dilution flow: when one forwarding rule replicates, another rule dies. To avoid the extinctive displacement of rules for suboptimal alternative paths, we keep topological information about the network's structure inside the nucleus which shields connectivity data from the fierce competition and path optimization. The role of the nucleus is to cultivate a variety of alternative paths and to periodically inject them into the forwarding engine. Even if suboptimal paths are squeezed out when another path suddenly becomes more attractive, the nucleus provides the initial rule (blueprint) for it, which can be re-instantiated later.



(a) (b)

Figure 6-8: (a) Node reaction vessels. (b) Experimental topology.

Assumptions

General: All nodes are static. Nodes advertise services (rather than reachability information). Zero state network is assumed regarding routing information. Every node only has knowledge of its immediate neighbourhood. Globally unique identification of nodes has been assumed.
Evaluation topology: As shown in Figure 6-8. Nodes 1, 5 and 7 have services to advertise. The remaining nodes provide only routing functionality.

Evaluation of Requirements

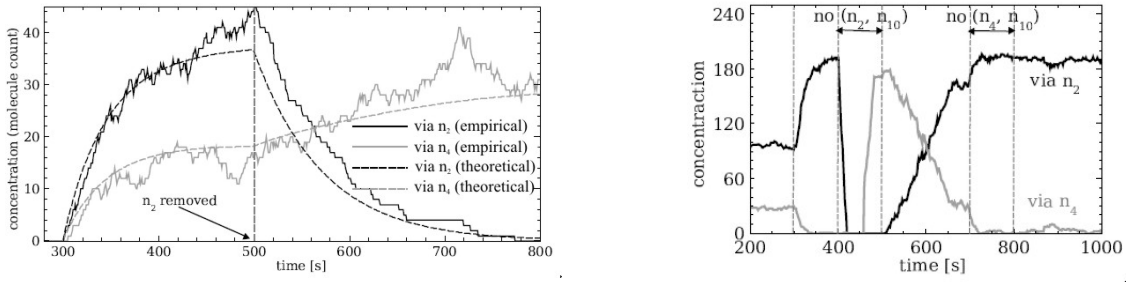
Information Gathering and Collection

- Situation awareness, detection of network conditions
- Detection of network anomalies
- Information about network resources

Figure 6-9 (a) shows what happens when node n_{10} joins the network at t = 300s and node n_2 leaves the network at t = 500s. At t = 300, the concentration of s_a molecules starts to rise in node n_{10} because it starts to broadcast route messages to its neighbours. Because of the dilution flow the concentration reaches a steady state in which there are more entries to

service s_a over the link to neighbor n_2 than n_4. When node n_2 disconnects at t = 500 s, the concentration of routing table entries for s_a slowly change to favour the remaining link via n_4.

We then arranged that the link (n_{10}, n_4) exhibits a packet loss probability of 20% while the remaining network is ideal. Data packets for s_a (i.e. traffic destined to service s_a) are generated at node n_{10} after t=300s after the beginning of the simulation. We proceed to examine what happens at node's n_{10} vessels. In Figure 6-9 (b) we see that after t = 300s, when data packets are injected, the forwarding rules over the loss-free link (via n_2) receive all acknowledgments and therefore win the competition. Consequently, rules over n_4 soon become extinct in the main vessel. This means that there is no alternative forwarding rule when the primary link (via n_2) is disconnected at t = 400s. In the next 100s no packets are forwarded anymore until finally a data packet is consumed by one of the rules over n_4 that are continuously re-generated by the riboquine (in the nucleus vessel). After this obstacle is overcome, the rule over n_4 quickly becomes stronger, because there is no competition. (The observed down time can be reduced by increasing the rate at which the riboquine injects new forwarding rules. This again highlights the importance of the separate nucleus which maintains the diversity of alternative paths).
Finally, at t = 500s, when we reconnected link (n_{10}, n_2), the lossless path quickly outperforms the alternative path over n_4 as desired. This shows that the attraction of the path over n_2 is much stronger.



(a) (b)

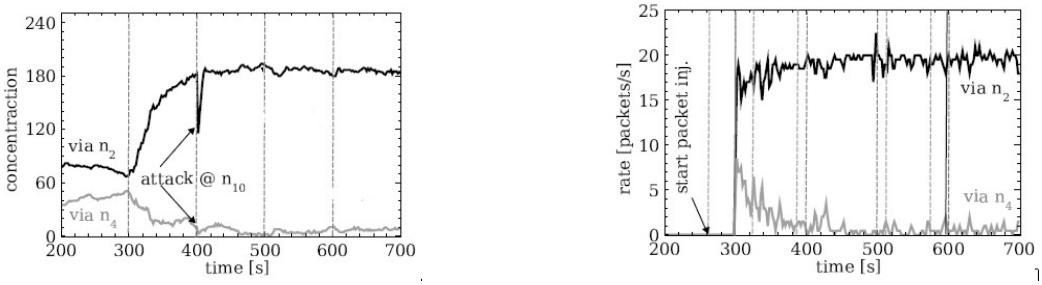
Figure 6-9: Concentration of molecules when nodes join and leave the network

Information Distribution and Node Collaboration (by Node Interaction)

- Common information model and protocols
- Information exchange in a standardised way
- Security framework, establishment of trust relationships

All nodes participating in the routing domain need to be able to abide to the information and execution model of chemical networking as introduced in D4.3 [24].
However, the routing protocol does not provide proactive protection from attacks, if malicious forwarding entries appear in the soup, unless they optimise the overall distributed system performance they eventually "die out" due to dilution. Maliciously removed forwarding rules, can be automatically re-created leading to recovery of the network in a re-active way: As riboquines periodically regenerate fresh rules, if the reliability of a path is re-established, the corresponding forwarding rules will increase allowing the re-utilisation of the path. This is shown in Figure 6-10(a) where at t = 400s, we randomly remove 50% of all molecules from both reaction vessels in node n_{10}. Even though the reduction is clearly visible in Figure 6-10 (b) the effect on data rate is hardly noticeable. In a long lasting attack data packets may start to be withheld due to the reduced number of forwarding rules. As soon as the attack is over the accumulated data packets will temporarily increase the reaction rate of the forwarding

quine due to the law of mass action, to compensate for the reduced transmission rate during the attack.



(a) (b)

Figure 6-10: Concentration and data rate when nodes were removed

Core Network Management Requirements

- Self adaptation of network components
- Management based on situation, policies and/or business objectives

Self-adaptation of the overall network (rather than its individual components) is an emergent characteristic in chemical networking, which is shown in all the experimentation we did, and is always situation driven. Policy based management is possible indirectly through rate control of influx and outflux (dilution), and through selection of policy based chem. reaction algorithms.

Performance of Network Management Mechanisms

- Self-adaptive behaviour optimizes available resources

The quines in the forwarding engine are not able to reproduce themselves immediately, since the required reward is deferred by the acknowledge mechanism. They may therefore become extinct, for example if the data packets or acknowledgement packets are lost/removed from the network. However, this extinction of code is desired, because it enables populations of successful forwarding quines to grow instead.

6.2.5 Ensuring INM Stability with Built-in Verification of Configuration Changes

Configuring an already running system is a difficult task as wrong settings can slow the system down or stop it completely. Our algorithm is meant to detect if new settings have a positive or a negative effect on the system.

The verification of applied configuration changes is based on a Markov chain. The system is aware of the current network conditions; each node takes real-time measurements of its surrounding network conditions. This data is used, together with already existing settings as an input for the change verification module. Based on setup thresholds the module computes the probabilities the system has of remaining stable with the new set of settings. That way, faulty settings are not applied and the system is kept alive. This allows for a fully distributed and adaptive approach to checking every setting applied to the network. The method is able to detect faulty changes regardless of the network size and type. The detection is fully distributed and takes place in all nodes; only the administrator is able to create a complete “picture” of the network. Upon detection, each node reports individually the detected problems so that the overhead is reduced. Using this distributed approach, configuration problems can get targeted in a rapid and reliable manner.

Assumptions

- Measurement of QoS parameters is available regardless of the network in case (size, type, etc.)

- Each node can obtain a snapshot of its surroundings (QoS information – link speed, link delay - and the currently applied settings)
- The system has been bootstrapped and is running

Evaluation of Requirements

Information Gathering and Collection

- Characteristics of devices
- Situation awareness, detection of network conditions

Low level information is needed for computing the effect of newly applied settings; this information is considered available. Device type is taken into account as the network setup strongly depends on this.

Information Distribution and Node Collaboration (by Node Interaction)

- Distribution of captured and collected information
- Distributed management and role based interaction

Detected problems are reported, automatically, only to the administrator; other data can be sent upon request from the administrator. Nodes share the detected problems to each other to check if the problem is only local or distributed. Errors are pinpointed as are the effects of it. The administrator receives a list of the affected nodes, the setting that caused the problem and the effect the new setting had on the network (which parameter degraded and how it happened).

Core Network Management Requirements

- Management based on situation, policies and/or business objectives
- Distributed Network Management architecture
- Make decisions and take actions

The management capabilities are distributed in each node. The built-in verification allows for a local management and thus reliability and security are increased throughout the whole network. Decisions regarding the applied settings are taken locally.

Performance of Network Management Mechanisms

- Reduction of computation load on each node by using distributed management
- Reduction of management information flow by using distributed management
- Quick switching of network wide behaviour
- Self-adaptive behaviour optimizes available resources

Resources are saved by using a distributed computing model (each node runs a simulation based on the current data; no history is needed and a minimum of network traffic is generated in order to take the necessary measurements). Scalability lies at the very base of the algorithm; it can be used in any sized network.

6.2.6 Self-adaptive QoS Management for VNets

In order to provide end-to-end QoS support inside a multi-domain architecture, we tested and validated a QoS paradigm for self-managing resources based on knowing the service requests and the network context called I-NAME (In-Network Autonomic Management Environment). I-NAME paradigm works in the self-organizing management plane as a

resource management function and offers services to VNet, which generates virtual networks, by negotiating the QoS parameters inside the established virtual resources. Moreover, if a VNet aggregates similar applications in the same virtual space, INM works for each particular application inside that space. Through simulation analysis and QoS parameters measurement in a multi-domain framework, we evaluate the ability of the I-NAME QoS proposed model: a) to select the best end-to-end path based on QoS profiles and b) to guarantee through adaptation the QoS parametric support for the selected path in a VNet. The scope of I-NAME is to enable the network entities with capabilities that automatically detects dynamically changing network configuration and reacts accordingly to the service requests. Distributed resource management places resource management functions into the network nodes, and FI paradigm must address the in-network management concept especially in the core network [38] [39].

Assumptions

The set of assumptions on which the I-NAME protocol it is based on are the following:

- The network topology consists of two types of access network segments connected through a core network infrastructure
- The links between nodes can be wired or wireless in the access network and wired in the core network
- The source node (SN) should be able to detect the flow of a specific application in order to send to the destination node (DN) the requested QoS profile for that data flow.

Evaluation of Requirements

Information Gathering and Collection

- Monitoring of lower layer information
- Characteristics of devices
- Information about network resources

To demonstrate I-NAME capabilities on providing end-to-end in-network resource management, we run time-critical test applications from SN to the DN under different network condition: with I-NAME support, over a Best-Effort network environment (i.e. IP) and based on QoS network layer classification. We compare each support model results in terms of average end-to-end transmission delay for the selected path in the network. In case of I-NAME, resource management support is based on the QoS Profiles message exchange between neighbour entities, on the path form the source to the destination [37].

Information Distribution and Node Collaboration (by node interaction)

- Distribution of captured and collected information
- Information exchange done in a standardized way
- Distributed management and role based interaction

The QoS parameters managed by the I-NAME QoS profiles inside the network are: (1) throughput, (2) delay and (3) jitter. Modelling a time-critical application, we imposed the maximum end-to-end accepted delay for a given application to 0.01 [s]. The application constrains are included in the messages containing the QoS profiles [37] and exchanged between neighbour nodes on the path form the SN to DN. The QoS profile negotiates in each network entity a set of QoS parameters that synchronize: (1) application requested QoS

parameter set, (2) parametric weights of each QoS parameter and (3) the entity capabilities on the path to carry the requested parameters.

Core Network Management Requirements

- Self-Adaptation of network components
- Distributed Network Management architecture
- Make decisions and take actions

Being a quantitative analysis, the ability of the I-NAME to manage the resources compared to QoS support mechanism on the existing access technologies is revealed by the following evaluation parameters: (1) end-to-end best path selection based on the use of QoS profiles in a dynamic network context, and (2) QoS parametric support guarantees for imposed application constrains.

Performance of Network Management Mechanisms

- Quick switching of network wide behaviour
- Self-adaptive behaviour optimizes available resources

Considering the effect of different number of transmitted packets per second over the average end-to-end delay for a constant packet size, I-NAME capability indicates corresponding modes of adaptation: through source fragmentation or through source code adaptation. Details have been presented in D-4.3. The benefits of I-NAME were proved running test scenarios in QualNet Developer 4.5. Considering the average end-to-end delay, Figure 6-11 shows for instance that I-NAME is able to maintain performances in a given range, compared to best-effort, QoS IP precedence 3 and 6 approaches.

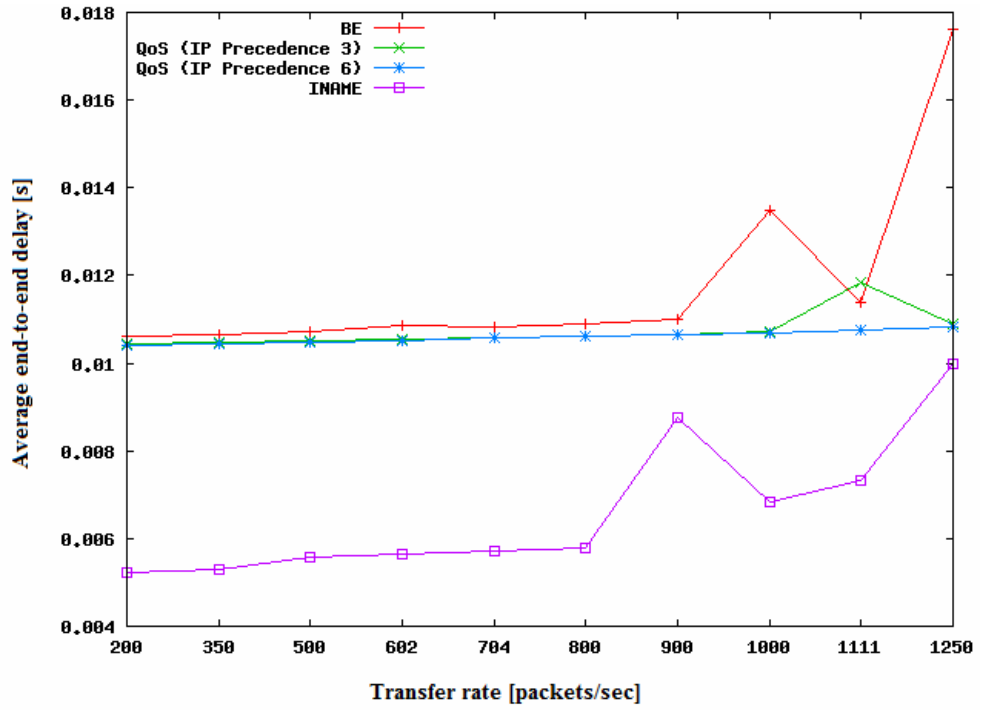


Figure 6-11: Evaluation of I-NAME advantages with respect to legacy approaches.

6.2.7 Emergent-behaviour-based Congestion Control

This mechanisms substitutes today's congestion control insides routers, which mainly influence TCP and lead to fairness problems. Based on the phase synchronisation of pulse-coupled oscillators, the filling level of queues along a path will be explored. The advantage of this method is that no explicit management actions or external control instances are needed to evaluate the congestion status along paths through the network. The mechanism is based on 4WARD Multipath-Routing – different paths are available to an ingress edge router and the related INM mechanism will decide, which path to use based on the congestion status. A more detailed description of this approach developed within 4WARD is described in the previous deliverable D4.3 [24] at section 9.4.3.

Assumptions

The following basic network assumptions are expected by the approach

- multiple alternative paths from ingress to egress available
- access to queue filling levels of each interface along a path
- exchange of status messages between routers

Evaluation of Requirements

Information Gathering and Collection

- Monitoring of lower layer information
- Situation awareness, detection of network conditions
- Information about network resources

This emergent congestion control mechanism is based on specific, lower layer information about the filling level of queues inside the network node along a path. Based on the emergent behaviour of the described algorithm, this leads to the detection of network conditions, namely the congestion status along a path. Note that there is no detailed information about a single, specific network resource (e.g. a queue of one node) available – due to the emergent behaviour it is always the information about a path via several nodes.

Information Distribution and Node Collaboration (by Node Interaction)

- Common Information model and protocols
- Distribution of captured and collected information
- Multicasting status and capabilities of nodes

The described mechanism is based on the exchange of status message between the routers. The messages (basically the synchronisation pulses) are of minimal semantics and can even be exchanged on lower protocol layers. The queue filling levels are bound to network interfaces and the related synchronisation pulse/message will be forwarded to next nodes along the chosen path.

Core Network Management Requirements

- Self-Management e.g. according to FCAPS model
- Distributed Network Management architecture
- Make decisions and take actions

The algorithm is intended to be exploited at the ingress nodes of a (sub-) network. Based on the congestion or queue filling status of different, available paths, an edge router decides on

selecting the best route for the next flow. Decision criteria might be QoS requirements of flows or overall load balancing inside the network, so many parts from the FCAPS model can be covered. There is no central component involved to keep the overall network status. The information is delivered to the appropriate edge nodes, which will then further relay this information along the expected routing path.

Performance of Network Management Mechanisms

- Reduction of computation load on each node by using distributed management
- Reduction of management information flow by using distributed management
- Reliability at least similar to centralized network management approach

Each router exchanges status messages with its neighbouring routers about its queue filling levels. Those messages can be exchanged via lower layers or even piggybacked and thus just require a minimal flow of management information. Moreover, all routers along a routing path just need to adapt their queue filling level parameter based on the highest distributed value which just requires minimal computational power. As the proposed approach continuously synchronizes and adapts its parameters itself it works reliable and in a fully distributed manner.

6.2.8 Self-Adaptive Routing in Wireless Multi-Hop Networks

An important feature of network nodes in wireless multi-hop networks is the ability of self-adapting behaviour, in terms of adjusting parameters such as metrics and protocols autonomously. A-HRP is a protocol that adaptively changes routing protocol and metric combinations for each path according to the network condition, such as node density or mobility. The proposed adaptation scheme also directly influences the topology knowledge of each node and is based on the goal to optimize the overall network throughput rather than optimizing the throughput of a single data flow. A more detailed description of this approach developed within 4WARD is described in D4.3 [24] at section 9.4.4.

Assumptions

A couple of basic assumptions to the network environment are expected for A-HRP in order to run properly: It is expected that all network nodes are

- Listening to the radio channel
- Willing to participate in ongoing communications of other nodes,
- Offering available resources in order to forward incoming traffic towards the destination.
- Able to not only act as a source or destination node, but must implement relaying functionality.

Evaluation of Requirements

Information Gathering and Collection

- Monitoring of lower layer information
- Situation Awareness, detection of network conditions
- Information about network resources
- Situation Awareness

All network nodes are continuously listening to the physical radio channel. They are overhearing information of neighbouring nodes such as link qualities, which will be used as

input to A-HRP. As each node has a multitude of lower layer information available, it can use this information as input in order to derive its current network situation, e.g. the mobility behaviour of a node.

Information Distribution and Node Collaboration (by Node Interaction)

- Common Information model and protocols
- Distribution of captured and collected information
- Distributed management and role based interaction

All nodes participating in communication are using the same protocol A-HRP and thus share the same data format. Gathered information, such as neighbouring nodes and Expected Transmission Count (ETX) [40] are distributed to neighbouring nodes that are within transmission range. Whether this information will be even further relayed via A-HRP up to the n^{th}-hop neighbourhood solely depends on the decision of the relaying nodes. Hence, the management of information distribution is realized in a decentralized manner.

Core Network Management Requirements

- Management based on situation, policies and/or business objectives
- Self-Management e.g. according to FCAPS model
- Self-Adaptation of network components
- Make decisions and take actions

A-HRP is an algorithm that enables network nodes to self-adapt their parameters, e.g. to change protocols and metrics. Based on a given network situation that is derived from information by overhearing the radio channel, each node makes its own decision on how these parameters should be adapted; trying to optimize the overall network throughput.

Data Communication Related Requirements

- Routing related functionality: Route discovery or self-rerouting

By adjusting parameters based on a given network situation, e.g. decreasing the broadcasting range of control information A-HRP also influences the flow of topology control information. As functionality like route discovery and self-rerouting depends on the knowledge of available routes to the destination, A-HRP is directly related to routing functionality.

Performance of Network Management Mechanisms

- Self-adaptive behaviour optimizes available resources
- Quick switching of network wide behaviour

To show that A-HRP is able to optimize available resources we have simulated the proposed algorithm in NS2 (Network Simulator 2) and compared the total amount of sent packets with metric/protocol combinations of AODV [43], OLSR [44], ETX [40 and Less Hop Count (which is the standard metric in today's Internet). The size of the network is fixed and set to 600m x 600m and the simulation time to 10 minutes. Three variable parameters, namely number of nodes, movement speed of a node, and the number of active connections, reflect a given network condition and are changed for each simulation run in order to reflect different network situations. Figure 6-12 shows the simulation results achieved in four different situations. The x-axis represents the number of nodes, the y-axis the number of successful transmitted packets and the caption of each figure reflects the selected parameters.

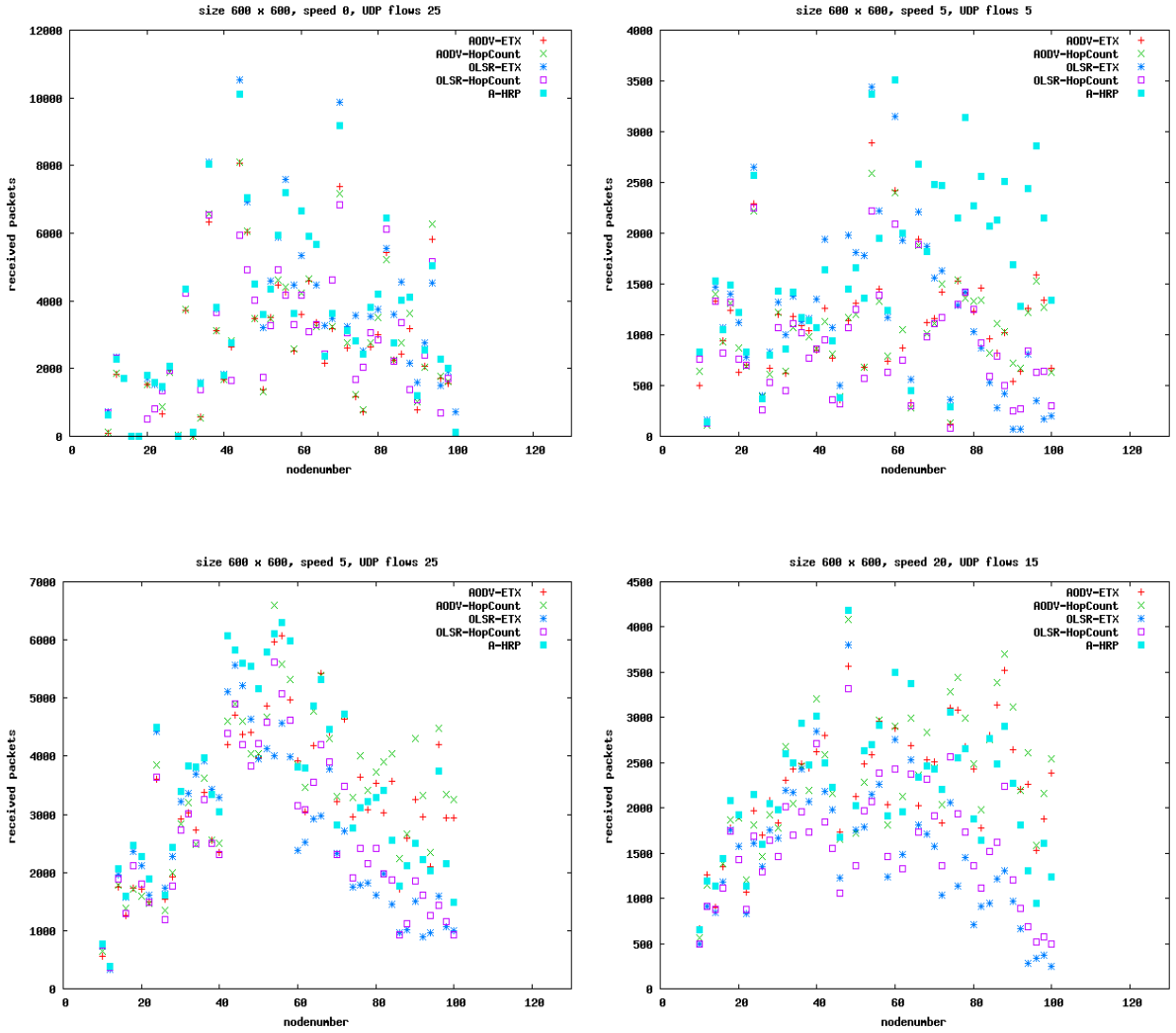


Figure 6-12: Simulation results of A-HRP

The results show that A-HRP is applicable for different situations. There are many situations where A-HRP outperforms the other approaches. There are also several situations where A-HRP performs in average, but there is no situation where it performs badly. Hence, overall A-HRP performs better than the other approaches that do not change their metrics.

7 Evaluation of INM with other WPs

This section briefly describes and evaluates collaborative work of WP4 with WP2, WP3, WP5 and WP6. The collaboration of WP1/4 is integrated in section 8 within the scope Business Values of INM.

7.1 Application of INM to NewAPC

Work was carried out with respect to mapping the architectural concepts originating from NewAPC with the output of the INM Framework task within INM. This work was then adopted by the Architecture Task Force (ATF). The work done within the scope of the ATF was not a specific evaluation but more so an overall mapping of architectural concepts, with investigations on how INM(and other WPs) related to the overall 4WARD System Model and on how non-functional requirements such as scalability and migration from legacy systems is addressed within 4WARD[1].

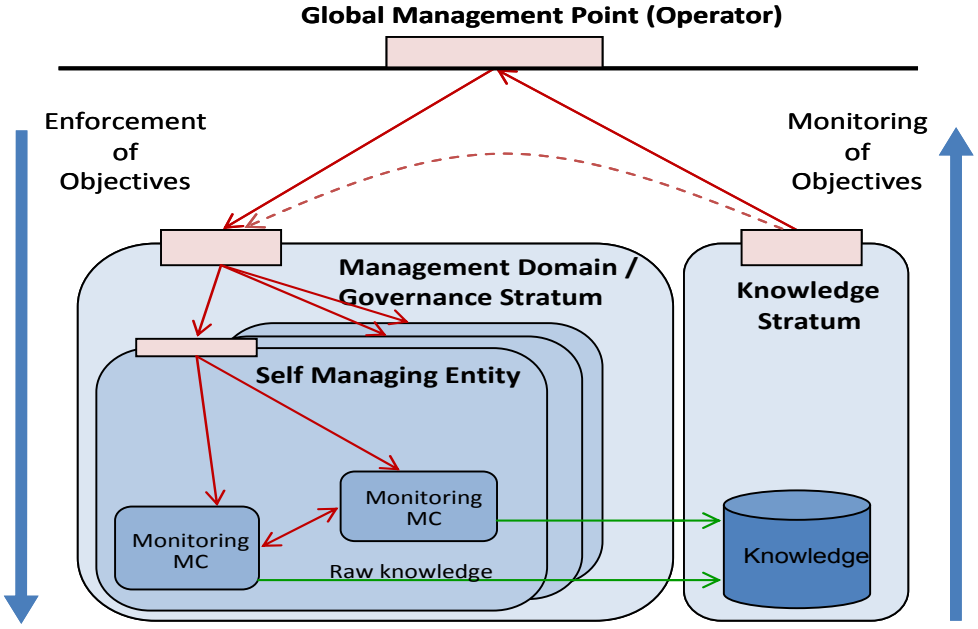


Figure 7-1: INM Relationship with Knowledge and Governance Strata

The 4WARD system model defines an 'IN Domain Management component' which in essence is the governance and knowledge strata relevant to a domain. These two types of strata are output of the NewAPC and represent how a domain will be governed and what knowledge is produced from within a domain respectively.

Figure 7-1 shows management objectives being pushed downwards through the governance stratum, into the SEs and eventually into multiple MCs which carry out the tasks in hand. The MCs in the figure could for example implement a monitoring algorithm. The output of the monitoring algorithm is in reality unprocessed data. This is fed into the knowledge stratum and reasoned upon and more high level knowledge generated. This knowledge is then used, possibly fed back into governance if some modifications or tweaking is necessary or displayed at a higher level as feedback on the objectives which an operator applied to the network. The GMP is the only management interface visible to the operator and provides the highest level of abstraction by means of objectives.

The description of the mapping between architectural components has been, in general, at a conceptual level but, there also has been a demo [27] which showed the Generic Aggregation

Protocol (GAP) running monitoring network congestion which fed into the Knowledge stratum. This in turn fed into the governance stratum and triggered a congestion control algorithm which attempted to handle congestion within the domain in question.

7.2 Application of INM to VNet

To evaluate the substrate node architecture, concepts, and the self-organizing model proposed in this work, we have to choose a substrate network model to be used. Despite the potential to gain a large market-share, network virtualization imposes significant enhancements over the current network models, such as new devices, transmissions management, etc. Given the problems on current models, we decided to work with a clean slate approach. Thus, we developed a network virtualization module based on Omnet++ simulator that uses a packet oriented transmission mechanism with traffic-shaping.

One important aspect of the implementation is the monitoring process, since the self-organizing algorithm depends on this process to make decisions. A two-stage monitoring process was defined. The first stage is always active and the size of data passing through the measurement points is the monitored information buffered. The second stage is periodically activated. Experimentally, we use intervals of 1.5 minutes. The monitored information on the first-stage buffer is summed and stored in a second-stage buffer. The self-organizing control loop uses the information of the second stage to determine the average amount of resources consumed within two self-organizing cycles. A sliding window keeps part of the information of the second-stage and the data from the first-stage is always erased. The sliding window helps to avoid that punctual high loads trigger constant reorganizations and lead the substrate network to an unstable state.

Network topologies and the initial mapping of the virtual IPTV networks are depicted in Figure 7-2. The substrate network is composed of 9 substrate nodes and each virtual IPTV network is composed of 3 virtual nodes (physically separated by 2 virtual pipes). Each virtual node in the figure is an IPTV Video Hub Office (VHO) able to store and transmit movies.

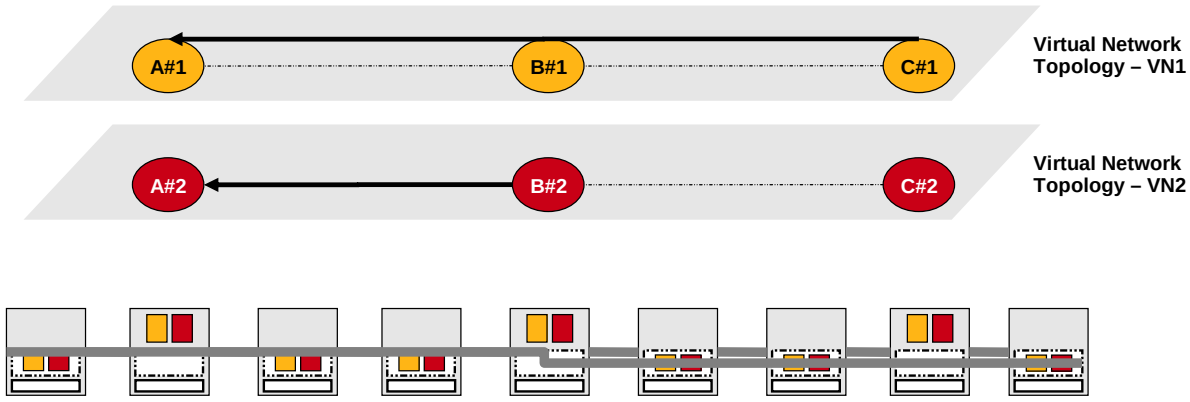


Figure 7-2: Evaluation Scenario

Two sets of flows are running inside the substrate network. The first one is associated to user's requests of the Virtual Network 1 (VN1). For VN1, the users connected to the VHO "VN1#N1" (inside node "B") are requesting movies that are associated to "VN1#N3" (inside node "H"). The movies are transmitted over the virtual link "VN1#L3", which is mapped to three substrate links "SL#1", "SL#8", and "SL#9". As depicted in Figure 7-2, the flows of VN1 start at node "H" and arrive at the node "B". The second set of flows is associated to the Virtual Network 2 (VN2). The users connected to the VHO "VN2#N1" (inside node "B") are requesting movies that are stored at "VN2#N2" (inside node "E"). The transmission occurs through the

virtual link “VN2#L1”, which is mapped to substrate links “SL#2”, “SL#3”, and “SL#4”. The flows of VN2 are originated at node “E” and arrive at node “B”. The bandwidth of each virtual link is 500 Mbits while the bandwidth of each substrate link is 1 Gbits. The size of the virtual storage associated to the VHOs of the virtual nodes is 50 GB, and the storage capacity of each substrate node is 100 GB. The size of the packets to be transmitted in the substrate links is fixed to 1 MB. The threshold to identify an overloaded link is the equivalent to 60% of the virtual link bandwidth. The amount of traffic inside virtual networks is mainly influenced by the number of movies requested by the users of the virtual networks. For this scenario, the request rate for each virtual network is fixed to 400 requests of movies per hour (400 req/h), and the interval between each request is given by an exponential distribution. The request rate is kept constant and active during the whole simulation. When a request arrives, the next action is the transmission of the movie. All movies in the experiments have the same size (4 GB).

The evaluation shows the efficiency of using the self organizing model in terms of spared network traffic. Using the scenario described above, almost 10 hours of user’s request were simulated, with the self-organizing cycle activated every 5 minutes. Traffic load of the substrate links and the average latency of the packets are measured every second stage monitoring interval (1.5 minutes).

We present in Figure 7-3, the sum of traffic loads of all substrate nodes of the scenario. Considering the scenario used on the evaluation, the total traffic load of the network is approximately 1.9 Gbits when the self-organizing model is disabled, and when the model is enabled it reaches the stable state using 1.2 Gbits. This means that 36.8% of the network resources of this scenario were spared when the self-organizing model is applied to manage the network resources.

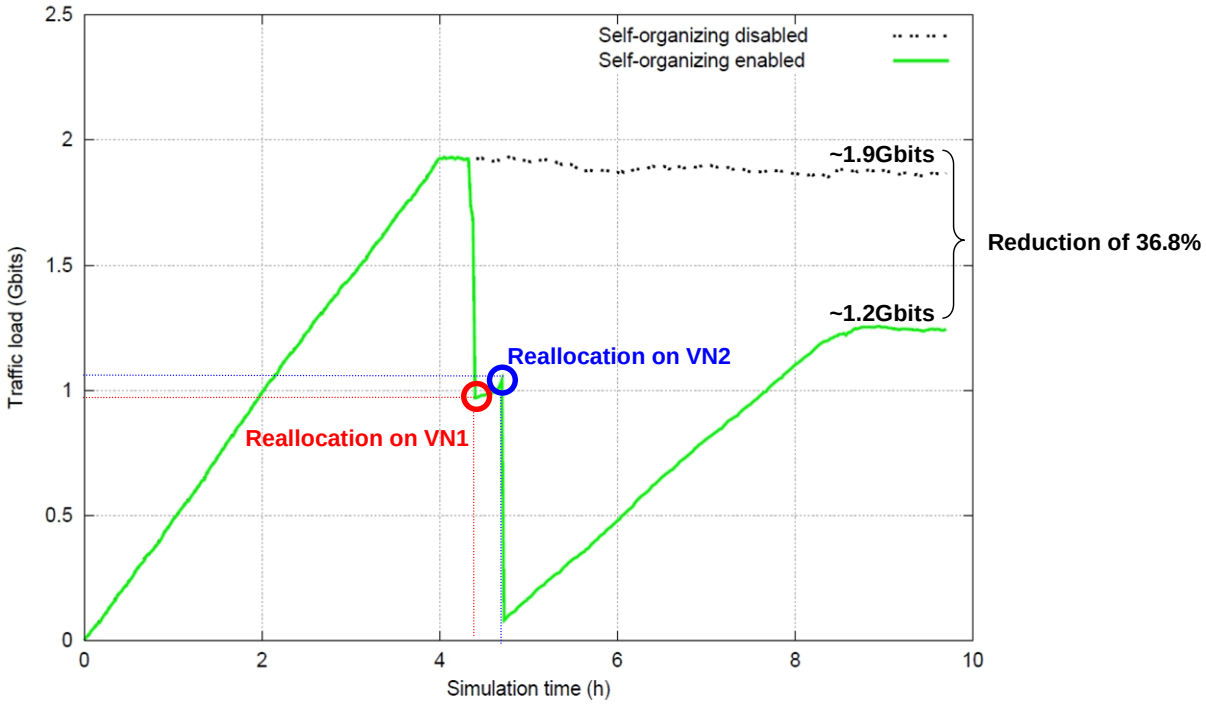


Figure 7-3: Overall Traffic Load

The next experiment shows the average latency of packets arrived in the destination virtual node of each virtual network within the monitoring interval (1.5 minutes). Figure 7-4 shows the packet latency measured for each virtual network. The packet latency for both virtual networks

is approximately 0.51 s when the self-organizing model is disabled, as illustrated in Figure 7-4 (left). The same average latency is observed in Figure 7-4 (right) until the first reorganization is executed. The average latency during the period of reorganization associated to the VN1 reaches 50.04 s, and during the reorganization related to VN2 it is approximately 50.84 s (not illustrated in figure). After the reorganization of the virtual resources the average latency remains stable around 0.48 s for both virtual networks. This value represents a reduction of 5.9% of the average latency as compared to the latency before the reorganization.

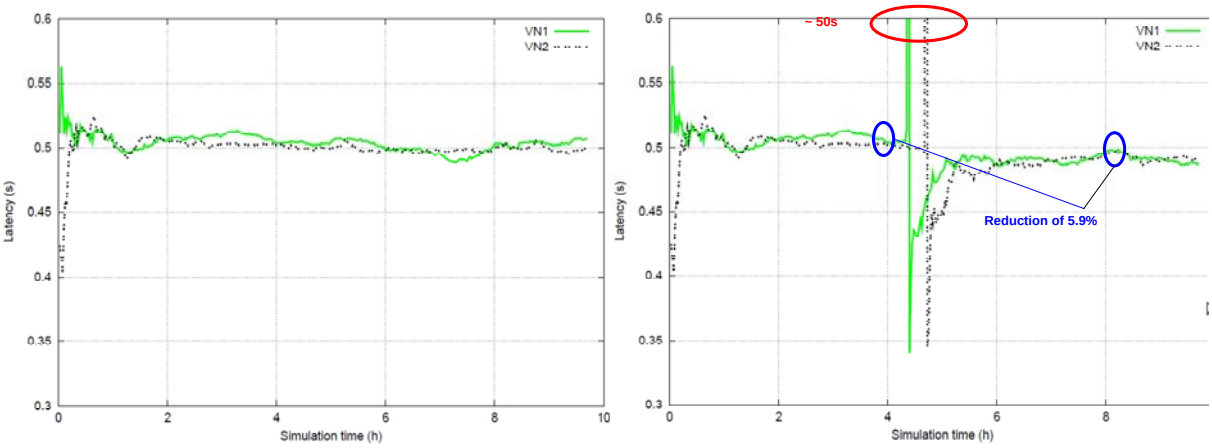


Figure 7-4: Packet Latency without self-organization (left) and with self-organization (right)

The self-organizing model can reduce the latency of the packets but the cost associated with this benefit is a high latency during a period of at least 1.5 min (around 50 s). This high latency might become a problem when the applications running inside the virtual network require strict QoS (Quality of Service) guarantees.

In summary, this work presents a distributed self-organizing model to manage the substrate resources in network virtualization using a specific module in INM. The main objective of this model is to manage the amount of network resources used during the lifetime of the virtual networks. The triggers of self-organizing actions are the local measurements and neighbor information. The experiments showed that the benefits in terms of reduction of traffic load are more expressive than the results in terms of latency. Moreover, the high latency observed during the reorganization process might reduce the range of types of virtual networks that comply with our self-organizing model.

7.3 Application of INM to GPs

The INM Framework developed in WP4 can nicely support many network functions, including those which are in the scope of WP5 (ForMux), i.e., the Generic Paths Architecture, data routing, network coding, resource sharing and mobility management. A joint group of researchers from WP4 and WP5 has formed TC45 task, with the main objective to identify synergies between both WPs. Because the Generic Path concepts and the INM Framework were developed in parallel, the activity related to the exploitation of common concepts started relatively late. As a natural consequence of this fact, in most cases the description of the exploitation of INM can be given at the conceptual level only.

In the context of exploitation, the INM functions have been divided into three groups:

- INM monitoring functions (including anomaly detection) for routing, network coding and resource management;
- advanced distributed congestion control schemes;

- INM self-adaptive and resource optimization mechanisms based on real-time monitoring.

Initial proposals of the usage of INM for ForMux (Forwarding and Multiplexing for Generic Paths) have been included in R-TC45.1. A more detailed analysis has identified 10 topics of common interest in which the exploitation of INM concepts are of common interest of researchers from both WPs. These topics are following:

- Generic Path Management Records for GP and INM interaction
- Cooperation and Coding Framework and the INM Framework
- INM assisted routing decisions
- Wireless network monitoring supporting self-adaptive routing and resource management in WMNs
- Resource Management for point-to-point communication in wireless networks
- Anomaly detection for point-to-point communication in wireless networks
- Anomaly detection and resource optimization
- Anomaly detection and multicast fast failure recovery
- Congestion control and multicast fast failure recovery
- INM cross-layer QoS used in network coding based GP combined with adaptive monitoring and anomaly detection

The above list is not an exhaustive one, but it just serves as an example of INM use cases. The presented topics will be described in more details in report R-TC45.2. In most cases this description is at the conceptual level. In the case of Network Coding GPs and INM Cross-Layer QoS, a demo implementation has been made. The obvious profit of the combined INM/GP research efforts is a clear separation of INM functions (monitoring, self-adaptation, congestion control) and the reusability for multiple purposes

7.4 Integration of INM approach into NetInf

The processes of designing the INM Framework and the NetInf architecture followed parallel tracks that were connected through the work in TC46. We highlight commonalities and differences between the INM Framework and the NetInf architecture, with the aim of evaluating the contribution of the INM Framework to adding self-management features to NetInf. As the INM Framework was not implemented by a NetInf prototype, it is impossible to provide a quantitative analysis regarding the extent in which the Framework evaluation criteria (scalability, robustness, reduced integration effort, reduction of complexity, ease of migration and maintainability) are to be reflected by the management functionality in NetInf. Instead, we provide a qualitative analysis.

Service orientation is defined as a cornerstone of the INM Framework. The NetInf architecture contains a service model [41], based on several components which are organised as services:

- the name resolution service and the storage service (together, they are known as the NetInf Dictionary)
- the Information Network Interface support for network transport services
- the event service
- the NetInf external services interface (for supporting additional functionality such as search services, for example)

Section 9.2 in D6.2 [41] reports on a qualitative analysis regarding the support of these services for INM Framework properties. The authors found that the NetInf Dictionary could

satisfy the self-knowledge and self-management properties due to its implementation using an enhanced Multiple-Distributed Hash Table (M-DHT) algorithm. Business goals related to the trade-off between the memory usage versus bandwidth allocation, to be handled via the Organisation interface, need to be reconciled with technical constraints since they affect the performance of the information fetch process [41]. Further to the remarks in that section, we could make the observations with respect to the Framework evaluation criteria:

- The MDHT–based implementation of the NetInf Dictionary is compatible to the scalability requirements of the INM Framework implementation. The MDHT-based overlay can easily scale to millions of devices and disseminate management information.
- The MDHT–based implementation of the NetInf Dictionary benefits from the self-healing properties of the DHT algorithm and is thus compatible with the robustness requirements of the INM Framework implementation.
- The simplicity of the NetInf service interfaces are in line with the reduction of integration efforts required by the INM Framework. The application of the co-design principle by embedding certain management functionality inherently in the NetInf Dictionary goes further along the lines of reducing the efforts of integration.
- The accounting mechanism described in [41] collects information about the network resources usage. It reuses the concept of Information Objects and stores the accounting data in a separate Dictionary overlay. The NetInf accounting mechanism provides the basis for implementing the accountability property of the INM Framework in a NetInf context.
- The INM Framework is strongly oriented towards services support through self-managing entities. The NetInf service model, presented in [41], distinguishes between internal and external NetInf services. The internal services are part of the NetInf machinery (dictionary, resolution event services). They interact through APIs and therefore do not need a contract, as specified by the INM Framework. In this respect, we could consider that the internal NetInf services implement some of the properties of the framework but they do no comply with the complete concept. However, external services (built by developers on top of NetInf) interact with the NetInf machinery through a contract expressed as an Information Object. Therefore, and taking into account the properties fulfilled by various parts of the NetInf machinery, it could be integrated as a self-managing entity in a network managed through the INM Framework.

[30] included a set of requirements for the INM use of NetInf technology for carrying network management information. The first requirement was related to the support for disseminating events asynchronously. NetInf defines an architectural element for the event service, although it does not specify any particular implementation for a notification mechanism. The INM framework does not specify how to disseminate events related to the parameters exchanged over the Organisation or Collaboration interfaces. As such, the flexibility of the NetInf architecture would allow INM functionality to utilise the push-based dissemination in publish-subscribe paradigms for some functionality (for example, data exchanged over the Organisation interface) while allowing for the use of adaptive push-pull schemes in other cases.

As opposed to generic information objects (IOs), network management-related IOs have a very well defined meaning (in most cases, the definitions are made within the framework of an international standard). As such, NetInf search functionality, which is build as an additional service on top of the NetInf machinery, could easily support searches for ranges of information relevant to the management. Such ranges of information are a requirement for using NetInf

IOs in the context of network management functionality [30]. The support for active query objects that would allow dynamic aggregation of information could be provided through the naming and metadata engines in NetInf. The last requirement, namely controlled access to management information, was left for further study in D6.2 [41].

8 Evaluation of Potential Business Values of INM

In the preceding chapters, technical aspects of the INM framework, schemes and algorithms for INM situation awareness and self-adaptation have been analyzed and evaluated separately. In addition, it has also been shown that INM concepts and methods can be used in other WPs of the 4WRAD project. These analysis and evaluation results are clear indications that the related INM aspects, schemes and algorithms are innovative in ideas and are effective in technical realizations.

This chapter goes further by trying to evaluate the INM approach as a whole. For a new approach like INM, it can only receive wide deployment and achieve large scale success in the long run if the approach as a whole can bring significant business values with it. That is, the long-term success of the approach will be dependent on the monetary benefits in addition to the technical merits. Above all, it is an interesting question to ask how much the INM approach can practically reduce the cost and the complexity of configuring and running networked services and thus achieve tangible business values for communication service providers (CSPs). Surely, such a question can only be answered in a concrete context where INM is used to implement salient management functionality so as to achieve technical effects on the targeted networking environments. For this purpose, this chapter investigates the potential business values of INM by using it to realize SON functionality in Long Term Evolution (LTE) networks.

8.1 Application of INM to Realizing SON for LTE

Long Term Evolution (LTE) has been introduced as an evolution of the 3GPP 3G wireless network standards [7]. LTE is expected to dominate the global market for mobile broadband over the next few years by offering major enhancements in speed, capacity and support for new services. With its support for a flat and efficient network architecture, LTE allows operators to deliver service-rich mobile broadband user experiences as well as to reduce their long term capital expenditure (CAPEX) and operational expenditure (OPEX). First commercial LTE networks have been rolled out recently. The success of LTE networks are considered to be highly important for both the operators and the vendors in the mobile communications business in the coming years.

Self-Organizing Network (SON) [1][4] is introduced as an enabling technology with LTE from the very beginning. In general, SON provides the means to automate the configuration, operation and optimization of cellular networks. Although SON is currently introduced with LTE, its scope can and should be extended to cover other technologies to cope with the management complexity in a heterogeneous networking environment. The functional scope of SON includes self-configuration, self-optimization and self-healing [5] [8].

It is evident that SON and INM share very similar motivations and goals. Both identify the needs to handle network complexity by bringing simplifications to the ways the networks are operated and managed. And, both SON and INM try to reduce human intervention to enhance efficiency and to avoid manual errors.

Principally, SON functions and schemes can be implemented either in a centralized manner, in a distributed manner or in a hybrid manner, dependent on the use cases [9]. Different SON use cases pose different requirements. Those requiring near real-time reactions in adaptations or optimizations of a management cycle (say, in the range of minutes or even seconds) are apparently more technically challenging to realize than those requiring less timely (say, in the range of hours or days). It is argued in the following that the realization of SON with the INM approach is one reasonable and natural way to achieve real-timeliness and efficiency.

The INM paradigm targets at a lean management plane in the network itself. A main objective is thus the design of management functions that are located close to the management services, in most of the cases co-located on the same network elements. As the target approach, the management functions should be co-designed with the services. Such an “embedding”/”inherent” characteristics of the INM paradigm is apparently a natural way to realize SON functionalities, which are generally centred around or are intertwined with user- and control-plane functions.

The INM framework supports management operations in the future networks by means of a highly distributed architecture. It advocates the realization of autonomous network elements with smart and distributed management decision making. The distributed architecture can thus leverage autonomous network elements with local SON decision making and reactions. This enables the implementation of fast SON capabilities within the network elements, without excluding the possibility of further more centralized coordination.

Thus, “embedding”, “distribution” and “autonomy” are the features needed by SON and their implementation by way of INM seems natural.

It is noted, however, that the utilization of INM principles for SON realization must initially remain in the general scope of the 3GPP management model [3]. Changes brought about by INM happen mainly by shifting and reshaping the activities of some management processes to a more autonomous and self-organizing manner. In a later stage, INM methods may contribute to the improvements of the existing management processes by simplifying some phases and/or extending other parts. More radical deployments of INM principles and methods may then be used if their utilizations turn out to be beneficial and successful.

Consequently, the combination of SON and INM can achieve good effects for operating and managing the LTE networks. In the following, the business values of this combination are investigated.

8.2 Areas of Business Values of SON-INM

Nowadays it is vital for a communication service provider (CSP) to have a rigid control of its network in view of total cost of ownership (TCO). TCO contains capital expenditure (CAPEX), i.e. the expenses for network solution and infrastructure, and operational expenditure (OPEX), i.e. the cost of keeping the network up and running. Sometimes, implementation expenditure (IMPEX), i.e. the cost of building the network, is listed separately.

As is argued above, upcoming networks such as LTE will require the realization of self-organisation features such as self-configuration, self-optimisation and self-healing. The "autonomy", "embedding" and "distribution" characteristics of the INM approach are helpful for the realization of SON functionality. A combination of SON and INM is beneficial for implementation easiness and management efficiency.

The business values of the combination of SON and INM can be identified in the areas of OPEX reduction, CAPEX/IMPEX reduction as well as revenue protection/increase, which results from improved network quality or enhanced user experiences. Some of them are indicated in Figure 8-1.

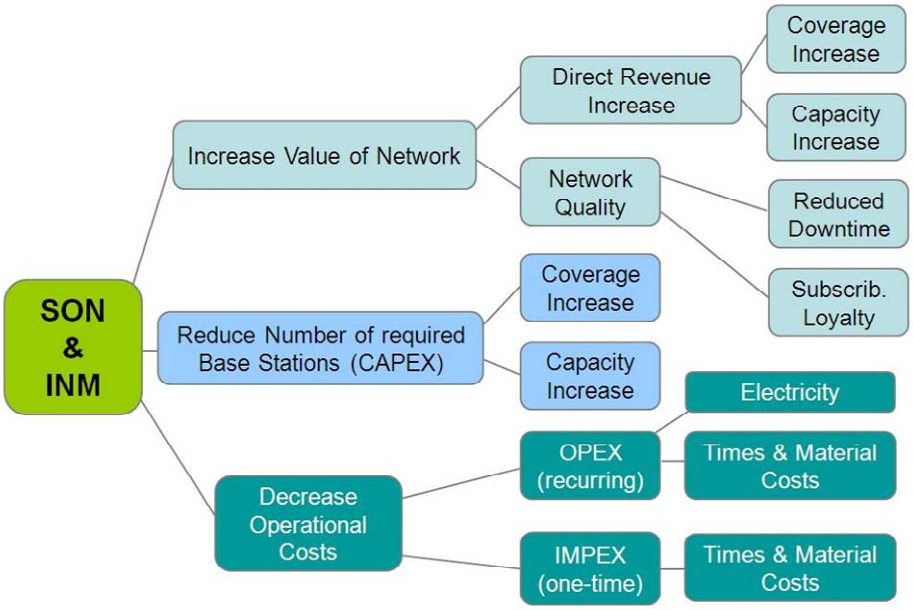


Figure 8-1: Benefits of SON-INM

8.3 Impact of SON-INM on OPEX and EBITDA

It is above all important for a CSP to control its OPEX due to its recurring nature and its sizable share in TCO. Earlier studies on public safety networks have shown that OPEX usually accounts for 50-80% of TCO over 10 years [10]. Similar statistics have been collected and shown for optical transport networks and mobile networks. Therefore, it is of extremely importance for a CSP to control its OPEX.

SON and INM can be used to improve many cost positions of OPEX, especially those related to network operations. The distribution of OPEX varies from developed markets to emerging markets or even from one operator to the next. Regardless of the variances, fixed and mobile operators normally spend more than 20% of their OPEX on network operations [11][12], which can be improved and optimized by SON-INM.

In 2007-2009, global spending by communication operators on OPEX exceeds $1000 billion per annum. Globally, $120 billion is spent on staffing network operations alone [11]. As such, SON-INM is addressing a market size which is quite significant for both operators and vendors.

For an operator, it usually matters more how much it earns than how high its revenue is. Simply put, revenue minus OPEX results into EBITDA (Earnings before Interest, Taxes, Depreciation, and Amortization). Unfortunately, continued OPEX increases have been observed by many mobile operators in recent years [13]. There are principally two ways of EBITDA improvements as is shown in Figure 8-2. Either one tries to reduce the OPEX or one tries to increase the revenue. As is explained in Figure 8-1, SON-INM can help in both ways.

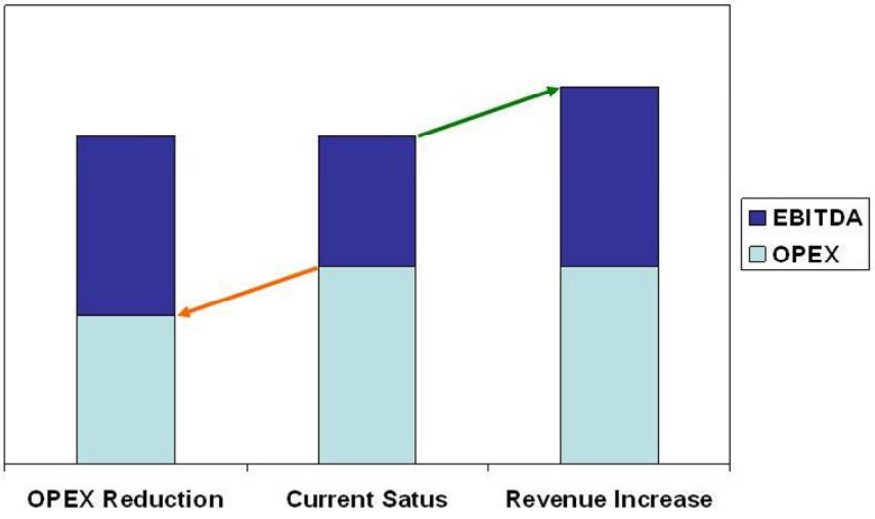


Figure 8-2: Alternatives of EBITDA Improvement

8.4 Quantifying OPEX and EBITDA Improvements

Sections 8.2 and 8.3 have shown that SON-INM can benefit LTE networks in many ways. This section concentrates on estimating their improvement on OPEX efficiency through a model quantification.

To get a detailed OPEX breakdown, one needs to list all the specific positions and proportions of OPEX-related spending. Again, the breakdown is dependent on where an operator resides and on the type of the operator. Although the breakdown is specific to each operator, typical compositions and elements have been analyzed [13] [11]. Figure 8-3 shows the model OPEX breakdown based on our research.

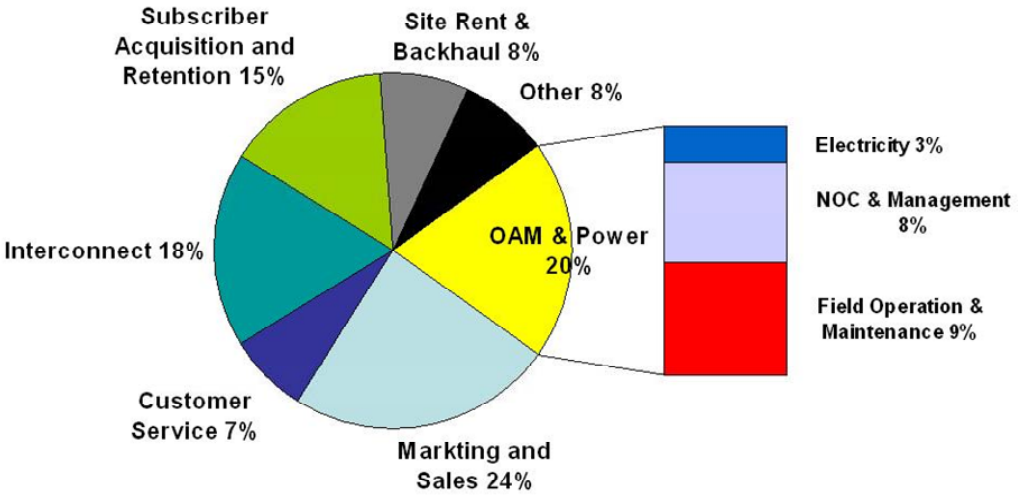


Figure 8-3: Targeted OPEX Positions by SON-INM

Three OPEX positions are highlighted in the figure above:

(a) With SON optimization, site visits and drive testing can be minimized. This and other SON use cases can lead to reduction in the cost of field operation and maintenance.

(b) With automated configurations and reactions, SON use cases can lead to reduction in the cost of operating NOC and other management tasks.

(c) Last but not least, SON use cases can reduce power consumption by switching on/off cells or adjusting the sending power of cells in clever ways.

Some analyses have shown that related management tasks can achieve improvements of 65% to 80% by SON [6][14]. As a result, OAM as a total will be able to achieve a significant cost reduction.

For the sake of a model computation, it is assumed that an example European mobile operator makes an annual revenue of 1 billion Euro with a current spending of 60% of it for OPEX. The EBITDA of the operator is thus 400 million Euros in the current constellation. In the model computation (with OPEX breakdown in Figure 8-3), it is estimated that the three OPEX positions in question constitute about 20% of the whole OPEX and can be directly optimized by SON-INM. As is shown in Figure 8-4, for an OAM reduction in the estimated range of 10%-40%, the corresponding OPEX decreases and becomes 588 to 552 million Euros. This leads to a corresponding EBITDA between 412 and 448 million Euro, which means a significant EBITDA improvement of 3%-12%.

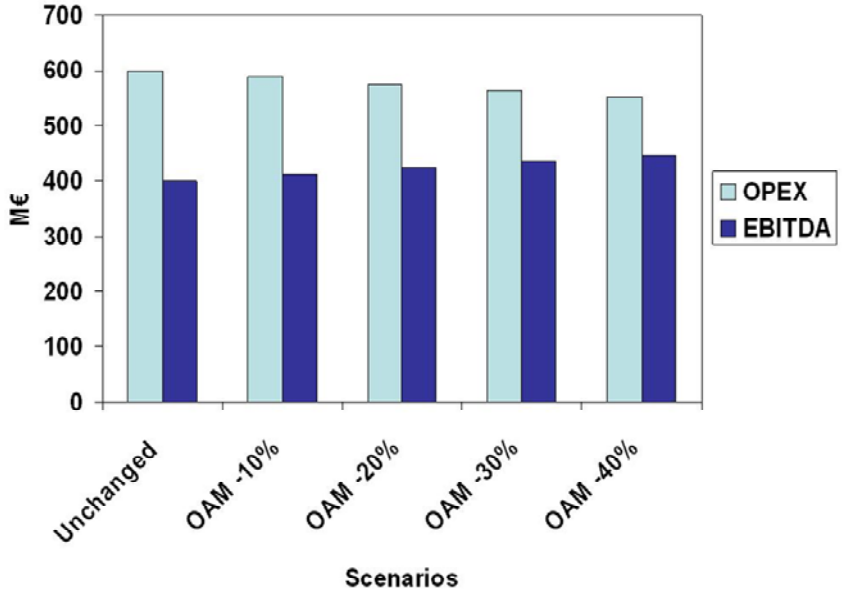


Figure 8-4: Comparison of OPEX/EBITDA Variances

In summary, LTE network is a prominent representative of the upcoming next generation mobile technologies. SON is deemed as a key success factor for the LTE networks. SON and INM share very similar motivations and goals by trying to simplify network operation and to reduce human intervention. The realization of SON with the INM approach will deploy a decentralized architecture with autonomous network elements. This is a straight-forward and natural way to achieve the real-timeliness and efficiency required by many SON functionalities. The combination of SON and INM has significant potential business values in the areas of OPEX reduction, CAPEX/IMPEX reduction as well as revenue protection/increase. Model quantifications have shown that SON-INM can reduce the cost of key OPEX positions and thus lead to direct EBITDA improvements.

As such, the INM approach as a whole is not only innovative in ideas and effective in technical realizations, but has clear potential for commercial success as well.

9 Conclusion

In this deliverable we evaluated the INM framework and distributed management algorithms regarding their suitability to enable INM features for the Future Internet. Additionally, evaluations for integrated approaches of INM and New Architectural Principles and Concepts (NewAPC), Virtual Networks (VNets), Generic Paths (GPs), and Network of Information (NetInf) were presented, while realising potential business incentives.

Inspired by a WP4 adapted V-model, the evaluation of INM was presented in a structured way and based on two closely related evaluation templates, one for the INM framework and another one for the distributed management algorithms.

To prove its suitability as enabler for the developed approaches the framework was evaluated based on the following general criteria

- Scalability of management operations

  Thanks to the organization interface the framework is able to hierarchically distribute management functionality in the system and thus to keep the system scalable. If needed, the framework also supports methods to guarantee accuracy of aggregated management information.

- Robustness

  The INM Framework is part of each Self-managing Entity (SE) and if used correctly applies the co-design principle as far as possible. It features the key properties self-monitoring, self-diagnosis, and decentralization and enables SEs to keep a managed system robust in means of maintaining consistency of performance under different conditions.

- Reduction of integration effort

  A reduced integration effort can be achieved due to modularity of Management Capabilities (MCs). These MCs can be positioned at different levels at the framework, such as inherent, integrated, separated or external. As each MC must implement an organization and collaboration interface, it is up to the designer to choose the most appropriate level of integration for an algorithm. Either way, the algorithm needs to publish its key outputs through a management objective allowing other system entities to subscribe to these key outputs.

- Reduction of complexity

  Complexity of the management system can be reduced by splitting functionality into smaller pieces and offering interfaces to interconnect them. One important interface here is the organization interface which abstracts some complexity in managing network functions by exposing high-level objectives in an aggregate form. Additionally, INM capabilities will be designed in compliance to the OSGi platform. Finally, the concept of co-design patterns supports the design of embedded, distributed, and large-scale management systems which aids to reduce the complexity as well.

The distributed management algorithms were evaluated against the functional requirement derived from the scenarios of D4.1 [22]. Table 9-1 shows a summary of Table 6-1, which visualized the addressed requirements for each algorithm. To be more precise, Table 9-1 shows for each specific requirement the frequency how often it is addressed by the distributed management algorithms.

Requirement	*Frequency*
Information Gathering and Collection	
Monitoring of lower layer information	**8**
Situation awareness, detection of network conditions	**13**
Detection of network anomalies	**5**
Characteristics of devices	**7**
Information about network resources	**14**
Information Distribution and Node Collaboration (by Node Interaction)	
Common Information model and protocols	**6**
Distribution of captured and collected information	**10**
Multicasting status and capabilities of nodes	**2**
Information exchange done in a standardized way	**7**
Distributed management and role based interaction	**12**
Security, establishment of trust relationships	**3**
Core Network Management Requirements	
Management based on situation, policies and/or business objectives	**7**
Self-Management e.g. according to FCAPS model	**6**
Self-Adaptation of network components	**9**
Distributed Network Management architecture	**8**
Make decisions and take actions	**8**
Special, Data Communication Related Requirements	
Traffic differentiation and handling	**2**
Routing related functionality: Route discovery or self-rerouting	**5**
Performance of Network Management Mechanisms	
Reduction of computation load on each node by using distributed management	**9**
Reduction of management information flow by using distributed management	**12**
Reliability at least similar to centralized network management approach	**8**
Substantially faster adaptation compared to centralized approach	**5**
Quick switching of network wide behaviour	**8**
Self-adaptive behaviour optimizes available resources	**11**

Table 9-1: Frequency of addressed requirements

As one can immediately see the developed algorithms within WP4 cover all aspects of the functional requirements derived from D4.1 [22]. Especially the typical tasks related to management, like information gathering, information processing and distribution and modification of system operation, are covered equally. However, some requirements are more often addressed as others. The following describes the reason of this for those requirements that have been addressed the most and the least often.

A handful of requirements have been addressed quite often (>10), which means that in average at least each second distributed management algorithm is addressing these requirements. These requirements are

- Situation awareness
- Detection of network conditions
- Information about network resources
- Distributed management and role based interaction

- Reliability at least similar to centralized network management approach
- Self-adaptive behaviour optimizes available resources

It is worth to notice that these requirements can be easily grouped into three topics, namely Situation Awareness, Self-Adaptation and distribution of network functionalities. It is quite obvious that these topics directly match with the main focus of the work of WP4 (distributing network functionality), T4.3 (Situation Awareness) and T4.4 (Self-Adaptation), which in turn means that WP4 addressed those requirements for INM that have been identified as the most important ones.

However, that does not mean that all requirements are addressed by each algorithm. In fact, the following three requirements have been just rarely/partially addressed

- Multicasting status and capabilities of nodes
- Security, establishment of trust and relationships
- Traffic differentiation and handling

Reasons for this weakness are versatile, though the most important one is a very specific problem space of these requirements. Other reasons are simplifications and assumptions that have been often made for the algorithms, such as a single domain or the usage of software that already integrates security concepts (e.g. OSGi and Java) as presented in the WP2/4 demonstrator. Finally, the requirements also overlap with topics that are closely related to work done by other WPs, e.g. “traffic differentiation and handling” and WP4 just provides needed information there, but does not further act on those.

The integrated approaches of INM were evaluated and briefly concluded in the following

- Application of INM to NewAPC

  INM architectural components are key building blocks in the realisation of the Knowledge and Governance Strata, as was shown through the WP2/4 demonstrator [25]. Also the INM algorithms, which are in essence management design patterns, can become very useful artefacts in the design repository which the NewAPC proposes [24].

- Application of INM to VNET

  Different simulation results showed improvements to VNET if a self-organizing model is applied that manages available network resources. For the described scenario the self-organizing model allows saving 36,8% of the network resources and reducing the average latency by 5,9%.

- Application of INM to GPs

  INM functions for GPs have been divided into three groups, namely INM monitoring functions, advanced distributed congestion control schemes, and INM self-adaptive and resource optimization mechanism. The evaluation of these INM functions are described in a separate report R-TC45.2

- Integration of INM approach into NetInf

  We concluded that a MDHT-based implementation of the NetInf Dictionary is compatible to the scalability and robustness requirements of the INM Framework as it easily scales to millions of devices and benefits from the self-healing properties of the DHT algorithm. Moreover, likewise the INM framework the NetInf service interfaces and the application of co-design principles by embedding management functionality inherently in the NetInf Dictionary also reduces the integration effort.

Finally, the Business Values showed that INM is a feasible way to realize Self-Organizing Networks (SON) in LTE, which can lead to reduction in the cost of field operation and maintenance and other management tasks. Analyses have shown that improvements of 65% to 80% can be achieved by SON.

Summing up the INM framework, distributed management algorithms and also the integrated management approaches that WP4 has been working on are feasible to cover needed aspects for an INM system of the Future Internet. The framework is able to support distributed management algorithms with needed interfaces that allow efficient embedding and deployment and enable them to support reliability and robustness. On the other hand-side the proposed management algorithms cover various aspects of network management with a main focus on enabling situation awareness, self-adaptation and distributing functionality. However, as it was not possible to focus the work on all technical aspects, but just on selected topics there are still problem spaces that need future work, such as distributed security and trust concepts or cross-domain management approaches. Additionally, it is important to notice, that the developed approaches have been evaluated conceptually, based on simulation results or a demonstrator but still need to be utilized in real experimental environments.

10 References

[1] "4WARD Deliverable D-0.5: Introduction and Overview of the 4WARD Technical Results", Martin Johnsson (editor), FP7-ICT-2007-1-216041-4WARD / D-0.5.

[2] NGMN Alliance, "NGMN Recommendation on SON and O&M Requirements", Dec. 2008.

[3] 3GPP 32.101, "Telecommunication management; Principles and high level requirements (Release 9)", V9.0.0, Sept. 2009.

[4] 3GPP 32.500, "Telecommunication Management; Self-Organizing Networks (SON); Concepts and requirements (Release 9)", V9.0.0, Dec. 2009.

[5] 3GPP 36.902, "E-UTRAN; Self-configuring and self-optimizing network use cases and solutions (Release 9)", V9.0.0, Sept. 2009.

[6] Nokia Siemens Networks, "Introducing Nokia Siemens Networks' SON Suite - an efficient and future-proof platform for SON", White Paper, 2009.

[7] UMTS Forum, "LTE Mobile Broadband Ecosystem: the Global Opportunity", UMTS Forum Report No. 42, June 2009.

[8] SOCRATES, "Use Cases for Self-Organising Networks", Deliverable D2.1, Mar. 2008.

[9] SOCRATES, "Review of use cases and framework", Deliverable D2.5, Mar. 2009.

[10] H. Juurakko, "Measuring and Managing the Total Cost of Ownership for TETRA Networks", Nokia, Nov. 2003.

[11] C. Mendler, "Evaluating the Benefits of Carrier Outsourcing", Presentation at Netevents APAC VIP Service Provider Summit, Yankee Group, May 2008.

[12] T. Smura (ed.), "Final Results on Economics of Converged Network and Service Environment", CELTIC ECOSYS Deliverable 22, Jan. 2007.

[13] Capgemini, "Quest for Margins: Operational Cost Strategies for Mobile Operators in Europe", 2009.

[14] Motorola, "LTE Operations and Maintenance Strategy - Using Self-Organizing Networks to Reduce OPEX", White Paper, 2009.

[15] R. Steinert and D. Gillblad, "Towards distributed and adaptive detection and localisation of network faults", AICT2010, Barcelona, Spain, May 2010. Accepted.

[16] M. Dam and R. Stadler. "A generic protocol for state aggregation". Proc. Radiovetenskap och Kommunikation (RVK), 2005.

[17] S. Krishnamurthy, J. Ardelius, E. Aurell, M. Dam, R. Stadler and F. Wuhib. "The Accuracy of Tree-based Counting in Dynamic Networks". Unpublished manuscript, 2010.

[18] S. Dolev, A. Israeli and S. Moran. "Self-stabilization of dynamic systems assuming only read/write atomicity". Distributed Computing, 7(1):3-16, 1993.

[19] A. Gonzalez Prieto, "Adaptive Real-time Monitoring for Large-scale Networked Systems," Ph.D. dissertation, Dept. Elect. Eng., Royal Institute of Technology, KTH, 2008.

[20] G. Kreitz, M. Dam, D. Wikström, "Practical private aggregation in large networks", Submitted for publication.

[21] R. Cohen and A. Landau, "Not All At Once! – A Generic Scheme for Estimating the Number of Affected Nodes While Avoiding Feedback Implosion", Infocom 2009 mini-conference, Rio di Janeiro, Brazil, Apr. 2009

[22] "4WARD Deliverable D-4.1: Definitions of Scenarios and Use Cases", Rudolf Roth, Fabian Wolff, Tanja Zseby (editors), FP7-ICT-2007-1-216041-4WARD / D-4.1.

[23] "4WARD Deliverable D-4.2: In-Network Management Concept", Giorgio Nunzi (editor), FP7-ICT-2007-1-216041-4WARD / D-4.2.

[24] "4WARD Deliverable D-4.3: In-Network Management Design", Alberto Gonzales (editor), FP7-ICT-2007-1-216041-4WARD / D-4.3.

[25] "4WARD Deliverable D-4.4: In-Network Management System Demonstrator", Giorgio Nunzi (editor), FP7-ICT-2007-1-216041-4WARD / D-4.4.

[26] "4WARD Deliverable D-5.5: Physical Layer Awareness", Filipe Cardoso (editor), FP7-ICT-2007-1-216041-4WARD / D-5.5.

[27] C. Moore, G. Goshal and MEJ Newman, "Exact solutions for models of evolving networks with addition and deletion of nodes", Physical Review E, 74(3):36121, 2006.

[28] L.Guardalben, V. Mirones, S. Sargento and P. Salvador, "A Cooperative Hide and Seek Discovery over In Network Management", 2nd IFIP/IEEE International Workshop on Management of the Future Internet (ManFi 2010), IEEE/IFIP Network Operations and Management Symposium, January 2010.

[29] F. Wuhib, M. Dam and R. Stadler, "A Gossiping Protocol for Detecting Global Threshold Crossings", IEEE Transactions on Network and Service Management, vol.7, no.1, pp.42-57, March 2010.

[30] Brunner, M.; Andersen, F.-U.; "Opportunities, Requirements and Challenges for Storing Network Management Information in a Decentralized Way," GLOBECOM Workshops, 2008 IEEE , vol., no., pp.1-5, Nov. 30 2008-Dec. 4 2008

[31] http://en.wikipedia.org/wiki/V-Model_%28software_development%29

[32] C. Alexander: A Pattern Language: Towns, Buildings, Construction. Oxford University Press, 1977.

[33] E. Gamma, R. Helm, R. Johnson, J. M. Vlissides: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1994.

[34] D. Dudkowski, M. Brunner, G. Nunzi, C. Mingardi, C. Foley M. Ponce de Leon, C. Meirosu, S. Engberg: Architectural Principles and Elements of In-Network Management. Mini-Conference IM'09. Long Island, NY, USA, 2009.

[35] D. Dudkowski: Co-Design Patterns for Embedded Network Management. In Proceedings of the 2009 Workshop on Re-Architecting the Internet (ReArch'09), Rome, Italy, December 2009.

[36] W. Martin and M. J. Russell. On the origins of cells. Philos. Trans. R. Soc. Lond. B. Biol. Sci., 358(1429), 2003

[37] Franzke M., et al., *In-Network Management Concept*, FP7-ICT-2007-1-216041, 4WARD/D-4.2, http://www.4ward-project.eu/index.php?id=64.

[38] Y.Ban, J.K.Choi, H.-S. Kim, *Efficient end-to-end QoS mechanism using egress node resource prediction in NGN network*, The 8th International Conference in Advanced Communication Technology (ICACT 2006), 20-22 February, 2006, Volume 1, pp. 480-483.

[39] M.S. Hemami, M.Pirhadi, A.I.Tabrizipoor, Analysis and optimization of resource control schemes in Next Generation Networks, First ITU-T Kaleidoscope Academic Conference - Innovations in NGN: Future Network and Services (K-INGN 2008), May, 2008, pp. 63-68.

[40] D. De Couto, D. Aguayo, J. Bicket, R. Morris, “A High Throughput Path Metric for Multi-Hop Wireless Routing,” ACM Mobicom Conference, September 2003.

[41] “4WARD Deliverable D-6.2: Second NetInf Architecture Description”, Matteo D’Ambrosio, Marco Marchisio, Vinicio Vercellone (editors), FP7-ICT-2007-1-216041-4WARD / D-6.2

[42] Cristián Varas, Thomas Hirsch, "Self Protection through Collaboration Using D-CAF: A Distributed Context-Aware Firewall," securware, pp.179-184, 2009 Third International Conference on Emerging Security Information, Systems and Technologies, 2009

[43] Ad hoc On-Demand Distance Vector (AODV) Routing: RFC 3561.

[44] Optimized Link State Routing Protocol (OLSR): RFC 3626.