A Framework for In-Network Management in Heterogeneous Future Communication Networks

Christopher Foley[1], Sasitharan Balasubramaniam[1], Eamonn Power[1], Miguel Ponce de Leon[1], Dmitri Botvich[1], Dominique Dudkowski[2], Giorgio Nunzi[2], and Chiara Mingardi[2]

[1] Telecommunications Software & Systems Group – Waterford Institute of Technology
Waterford, Ireland
{ccfoley, sasib, epower, miguelpdl, dbotvich}@tssg.org
[2] NEC Laboratories Europe, Network Research Division, Heidelberg, Germany
{dominique.dudkowski, giorgio.nunzi, chiara.mingardi}@nw.neclab.eu

Abstract. Future communication networks will be composed of a diversity of highly heterogeneous network variants, ranging from energy constrained wireless sensor networks to large-scale wide area networks. The fact that the size and complexity of such networks will experience tremendous growth will eventually render existing traditional network management paradigms unfeasible. We propose the radically new paradigm of *in-network management*, which targets the embedding of self-management capabilities deep inside the network nodes. In this paper, we focus on our *framework for in-network management*, which allows management logic to be embedded and executed within network nodes. Based on a specific use-case of bio-inspired network management, we demonstrate how our framework can be exploited in a network failure scenario using quorum sensing and chemotaxis.

Keywords: in-network management, bio-inspired self-management

1 Introduction

A new management paradigm for the Future Internet is being developed within the 4WARD project, driven by a European consortium under the FP7 research program. The proposed “In-Network Management” (INM) paradigm leverages on the high integration of management functions with the network components: management functions are seen as embedded capabilities, which differ radically from the traditional design and deployment of management functions as add-on features. The benefits range from increased network autonomicity to reduced cost of integration.

While embedding management capabilities in the network itself is a promising approach, this level of integration requires well defined functions in the network element: the question addressed by this paper is whether the INM paradigm can be deployed maintaining a certain level of generality and extensibility, which are two necessary properties of complex management systems.

INM does not bring incremental improvements to existing network functionalities like much of the autonomic community have been following. Instead, INM is pursuing a *clean slate design* for the Future Internet in an attempt to radically redesign today’s networks based on novel principles. With this respect, 4WARD follows other similar initiatives of different research communities, like [15, 16]. Nevertheless, the INM paradigm appears as a novel paradigm not covered by these initiatives.

This paper introduces our current state of a framework for INM, which follows a clean-slate design approach. It seeks to support the management tasks of the future Internet, from the deployment to the running of management functions as embedded management capabilities in the network, to their interaction and collaboration. The framework will enable network operators to have greater knowledge, make the networks easier to manage, and lessen the workload on operators to spend increased time working with the Operation and Support Systems.

Chapter 2 introduces our INM paradigm. The basic framework components are discussed in Chapter 3. In Chapter 4, we provide a detailed case study based on bio-inspired networks to show how the INM framework can be exploited. Related work is discussed in Chapter 5, before we conclude in Chapter 6.

2 The In-Network Management Paradigm

In-network management is a novel paradigm that proposes a new approach to perform management operations in future networks. We recognize that the bottleneck of traditional solutions is structural to the paradigm of traditional approaches, where management operations are normally seen as “add-on” features of network devices: first network functions are deployed in the devices, then management functions are added to perform FCAPS operations. Reduced scalability, high integration costs, lack of automation are the first main shortcomings.

The paradigm of in-network management assumes embedded management capabilities, where several autonomous components with management capabilities inside a network element allow for a flexible composition within the same or between different devices. Consequently, management operations become strongly localized and different network elements interact based on peer-to-peer techniques.

As the first design milestone, a new definition of a framework is required to support the newly designed management capabilities. While embedded management capabilities enable highly localized management functions, a framework is in fact needed to compose management functionalities and make them work on a large scale. The objectives of the INM framework will proceed in three directions: keep a low footprint of the embedded functions in the network elements, enable discovery mechanisms and support dynamic deployment of management capabilities.

The first architectural element is to which extent we can push management capabilities inside a component. For this reason, we defined different degrees of embedding, which help in guiding the definition of management functions as embedded management capabilities. The degrees are defined as follows: **Inherent** management capabilities are an integral and inseparable part of a functional component’s logic which cannot be altered. **Integrated** management capabilities are internal to a functional component, but separable from a functional component's logic. They allow for paradigms to load management capabilities into a functional component. **External** management capabilities are located on another node.

This categorization is an instrument to deploy the INM paradigm while maintaining generality and extensibility. For example, the inherent model can be used to support protocol-specific congestion control mechanisms and to integrate them into the management plane of INM, as depicted in Fig. 1 (left). Today, TCP and SCTP have an inherent congestion control mechanism: a fault management tool requires additional read operations to monitor several counters from the nodes. We can see the difference between the traditional approach, where separated management functions must be put in place after network deployment, contrasted by the INM paradigm, which is based on a new architecture of embedded management capabilities.

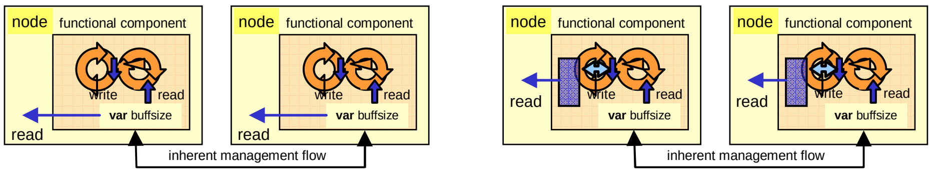

Fig. 1. Example of inherent (left) and integrated (right) management capabilities.

The inherent model introduces a well defined interface to access the embedded capabilities. The integrated model maintains extensibility of a component’s capabilities. As an example, we show in Fig. 1 (right) how this model can be used.

We believe that this categorization enables a lean and yet extensible software deployment. The next section shows how embedded capabilities can be deployed to perform more complex management operations.

3 Framework for In-Network Management

The diversity of node types which the proposed framework addresses is extensive. The design of the framework architecture must satisfy the requirements coming from both a small sensor node and also a powerful server node. Therefore it must be modular, in that the most fundamental components or artefacts can be easily exposed and also that the more high level components can be easily added and extended.

Fig. 2 shows the components which make up the framework architecture. The architecture defines a runtime environment in which functional components may be deployed. Within the runtime environment a number of levels of abstraction are defined that are relative to the amount of capabilities which a functional component makes accessible to other components and applications. The architecture also defines specific services which aid the running of functional components and applications. They are utilities within the architecture and are named InNetMgmt Services.

The InNetMgmt Kernel is a privileged area within the architecture where components or services which run here are protected from application interference, in that access to artefacts inside of the InNetMgmt Kernel is restricted. Security measures exist when accessing this area of the architecture.

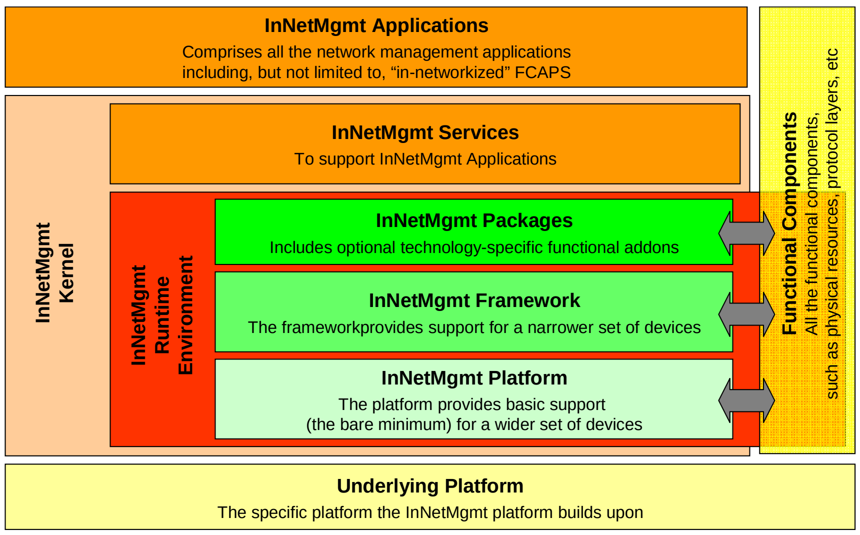
InNetMgmt Applications
Comprises all the network management applications including, but not limited to, "in-networkized" FCAPS
InNetMgmt Kernel
InNetMgmt Services
To support InNetMgmt Applications
InNetMgmt Runtime Environment
InNetMgmt Packages
Includes optional technology-specific functional addons
InNetMgmt Framework
The frameworkprovides support for a narrower set of devices
InNetMgmt Platform
The platform provides basic support (the bare minimum) for a wider set of devices
Functional Components
All the functional components, such as physical resources, protocol layers, etc
Underlying Platform
The specific platform the InNetMgmt platform builds upon

Fig. 2. High-level node architecture.

3.1 Framework Components

The **InNetMgmt Runtime Environment** is a container in which functional components and, InNetMgmt Services can execute. The Runtime Environment contains three layers or levels of abstraction: the InNetMgmt Platform, the InNetMgmt Framework and the InNetMgmt Packages. These three levels are abstractions of the most fundamental libraries and capabilities to make InNetMgmt components and applications run. Additional packages are needed for more advanced functionality supported by specific device types.

Functional Components are logical entities inside a node that may contain some management capabilities or the ability to link to management capabilities in order to participate in the execution of a management function or algorithm. A functional component can be anything such as a device driver, a protocol layer, a service, or part of a service. Functional components can be dedicated management components in that their primary purpose is to execute dedicated management logic. They can also exist with just functional logic and not have any explicit management logic.

InNetMgmt Services are utilities within the framework which can take a number of forms and perform a number of functions. Their primary tasks is to provide fundamental support for InNetMgmt functionality, e.g. a command mediation service which provides a mechanism which applications can use to issue commands to and receive responses from functional components. An InNetMgmt service could be used as an alarm/event publication facility which could be availed of by applications. The developed InNetMgmt services will be relative to the capabilities of the node itself and to the features which is supported by the Runtime Execution Environment.

Another task which may be assigned to an InNetMgmt Service is the management of the functional components. This includes the starting, stopping and management of dependencies between components.

The InNetMgmt Services have governance over the node resources in that they have primary ownership of them, but this may be outsourced to a functional component if the necessity arises. InNetMgmt services assist developers and network operators in the deployment of in-network management.

3.2 Functional Component Interface Types

Components interact through a number of interfaces, all depicted in Fig 3. All components will expose a supervision interface. This will primarily be used for the purpose of starting, stopping and monitoring of the component. The service interface is used to expose the functionality relative to the domain which it is representative of. This interface is used by other functional components and also by applications running on the same node or running remotely. The management interface is central to the INM concept in that it allows for the exchange of management information between functional components. The management algorithm is embedded into the functional component and this algorithm could be distributed over a number of nodes. The management interface allows communication across multiple nodes as this is a key requirement to network management.

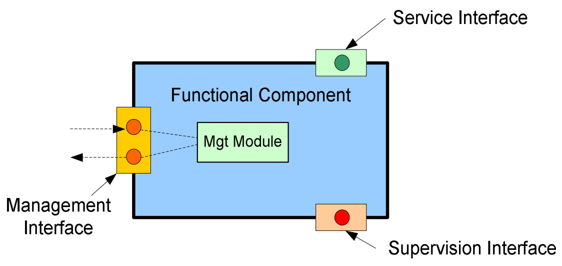

Fig. 3. Functional Component Interface Types

3.3 Management Function Calls

Whether a component or utility service resides in user or utility space depends on the functionality it contains. The fact that management logic can be running in a component in the kernel of a networked node is one of the key features of the INM paradigm. This gives the possibility to potentially analyze packets passing through the node and inject new or modified packets back onto the network. In-Network management does not just provide high abstractions from a network management level, it enables management at as low a level as possible. Because of the traversal of space, the framework will provide a bridge which will link the management of components and services across both user and kernel space.

4 Case Study

Some bio-inspired techniques applied to routing were chosen as the case study because it highlights the potential which exists by adopting the INM paradigm. The proposed framework when complete can avail of these techniques and algorithms as they can be embedded right down at a very low level within the node.

Our proposed bio-inspired techniques that have been applied to this case study are based on our biological framework for autonomic communications [11] [9]. In particular for this case study, we have employed two specific bio-inspired techniques, including: (i) Quorum Sensing, and (ii) Chemotaxis.

1. *Quorum sensing* [8] is a mechanism used by cells to coordinate in a distributed fashion to perform specific functionalities. The process mimics reaction-diffusion [1] mechanisms of cell self-organisation, whereby cells emit chemicals to the neighbouring cells. Each neighbouring cell will evaluate the chemical concentration and determine how much of its own chemical should be emitted into the environment.
2. *Chemotaxis* is a specific mechanism of mobility used by micro-organisms, which works through attraction of chemical field formed within the environment. There are two types of chemotaxis, which includes positive and negative chemotaxis. Positive chemotaxis allows the microorganism to get attracted to a food source, for example, while a negative chemotaxis pulls the micro-organism from a specific source (e.g. Poisonous source).

Mapping of Bio-Inspired Techniques

In this section we will describe how the two bio-inspired mechanisms described above will map to the INM resource management process of mesh networks. Large-scale networks with a large number of nodes will require an efficient mechanism to support routing and resource management for QoS (Quality of Service), which in turn will maximise revenues of the operator. Based on our bio-inspired frameworks, the two techniques have been applied to other networking concepts; chemotaxis to routing in core wired networks [11] and quorum sensing to ad hoc social networking applications [10]. In this case study, we employ the two techniques in a similar fashion to routing in a network failure scenario. An example of each of the two mechanisms is illustrated in Fig. 4.

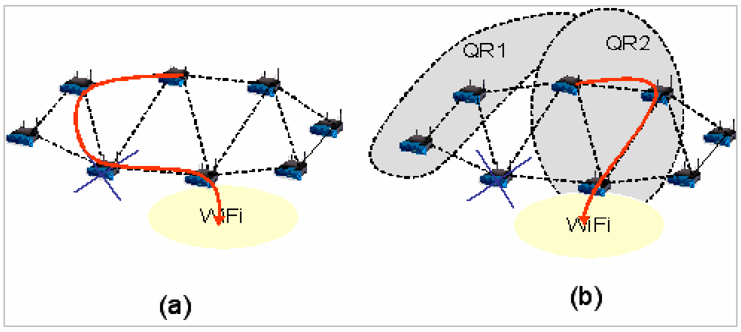

Fig. 4. Illustration of Chemotaxis (a) and Quorum Sensing (b) Techniques.

The chemotaxis process allows each node to translate the resource capacity of the node in terms of gradient G_i (i – node). As the route moves from one node to the next node, the route moves along the highest gradient from source to destination. This creates a hop-by-hop route discovery effect as shown in Fig 4 (a). In the event of a node failure, the nodes will collaborate with each other to invoke traffic prioritisation (this is due to the fact that there is not enough capacity to support all diverted traffic). The collaboration will be based on the revenue objective of the operators and the dropping of certain traffic types to maintain a certain level of revenue. The negotiation process and collaboration is based on the quorum sensing mechanism. As shown in Fig 4 (b), two quorum sensing regions are formed in the network (QR1 and QR2, based on traffic capacity capability). Since QR2 provides a larger amount of capacity for the diverted traffic, the gradient is higher, leading the diverted traffic to move to QR2.

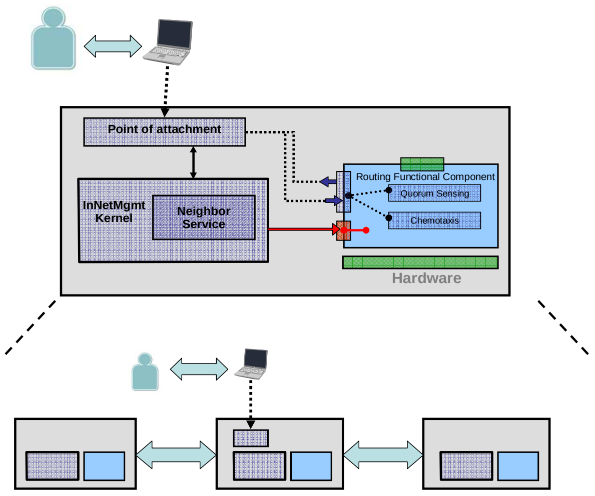

Fig. 5. Inclusion of embedded bio-inspired capabilities with an operator's objectives.

It is interesting to verify how these embedded capabilities can be actually accessed and controlled by an operator. While communication between nodes can be accomplished with peer-to-peer techniques[1], the composition of embedded capabilities and their linkage to an operator's objectives is a crucial design aspect of INM. Following the architecture principles of Fig 2, a point of attachment with the operator can be deployed as an INM application, attached to the kernel.

This application would then be responsible to retrieve the objectives and disseminate them in the network. Only one node is required to deploy this application, because each node is responsible to disseminate the behaviour inside the network in a P2P fashion. It should be noted that this is different from other approaches, where a policy server is required to translate objectives into a set of atomic operations and enforce them to all the devices.

Fig. 5 shows additionally how the bio-inspired techniques of Chemotaxis and Quorum Sensing would be deployed within the INM framework as embedded management capabilities. The primary interface which would be used by the modules is the management interface. It is through this interface that the gradient will be diffused between nodes and also the negotiation and collaboration process which is necessary to realize both algorithms.

Since the presented use case deals only with routing objectives, an interesting aspect and open question for the future architecture is to which extent different embedded management domains can be composed together. When different use cases are deployed together, translation of the objectives for different embedded capabilities becomes more complex.

Scenario Description

For the purpose of testing the assertions related to bio-inspired mechanism, we created and carried out tests on a wireless mesh network. We considered the network performance in a scenario to manage traffic flowing over a network and we compared conventional network technologies against a bio-inspired optimization technique.

The network represents that of an operator providing services to connected clients. The client pays for a data service and value-added services from the operator such as multimedia (on-demand TV). We refer to a wireless mesh network, because this is the technology requiring advanced management capabilities in future deployments. The same considerations are also valid for fixed networks, like routing in an optical ring within a metropolitan distribution network. We assume two QoS levels: (i) Multimedia (higher revenue), (ii) Data (lower revenue).

The test bed is illustrated in the Fig. 6. Its composition is designed to demonstrate the scenario with the minimum complexity, while demonstrating steady state conditions after a fault. For this reason, while we are testing on a mesh network, the link configuration implies that node A and C must communicate via the intermediary nodes of B and D.

[1] Following the INM paradigm, it is under discussion in 4WARD whether such p2p communication should occur in-band or out-band.

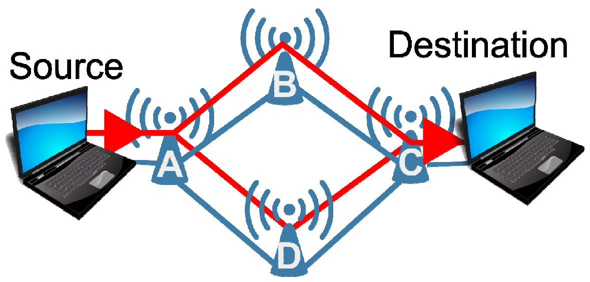

Fig. 6. Test bed setup for the reference scenario.

The Source was connected to node A, and the Destination to node C. Both data streams commenced at the Source to the Destination over the wireless network.

An Optimized Link State Routing (OLSR) stack was used to configure the routes on the wireless nodes. The stack is based on the Python scripting language, which allows us to quickly embed management capabilities.

```
  *Rule*:

         {CommonName
  "NodeDown"}

         Events
            {Name
  "ConnectivityChangeEvent",
          ParameterList
  "NodeIdentifier",
  "ChangeType"}

         Conditions
            {Name
  "LoseNodeConnnectivity",
            If ChangeType ==
  Disconnected}

       Actions
          {Name
"ApplyTrafficWeightingForRoute
Capacity",

       QueryRouteCapacity
              Params Source,
Destination}
```

Fig. 7. Bio-inspired mechanism triggered through a Rule.

In fact, the bio-inspired fault recovery is triggered through a node down event, described in Fig. 7. This rule was implemented via the Python scripting language.

The test-case shows how a management capability can be embedded inside a node. The parameters to be configured (the TrafficWeights of Fig 8) can be accessed through the internal management interface supported by the architecture and used by any application deployed on top, like a translator of objectives.

The steps for testing each scenario are detailed below. Phase 1 represents the normal operation of the network, where traffic uses the capacity of the entire network to allow the data streams to cross, Fig. 6. In phase 2, Node D encounters a failure; therefore a portion of the traffic must be diverted to node B.

Each node's physical hardware is a Ubuntu Linux 7.10 Server-based mesh node, consisting of the following: Mini ITX PC with two Atheros-based [17] wireless network interfaces. For the OLSR testing, the Python-based pyOLSR stack was used.

Results

The results in Fig. 8 show the traffic trace that is collected at the destination. In the conventional scenario using the OLSR mechanism, the traffic re-routes with no awareness of traffic prioritisation. Fig 8 shows the OLSR-based routing traffic has a reduction of total bandwidth, not taking into consideration the different traffic types, multimedia and data. Fig. 9 shows the effect of the bio-inspired quorum sensing mechanism. As depicted, the quorum sensing technique diverts the correct proportion of multimedia traffic through to node D, after the failure

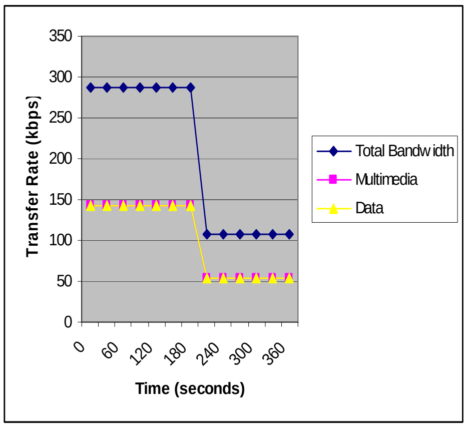

Fig. 8. Conventional Scenario

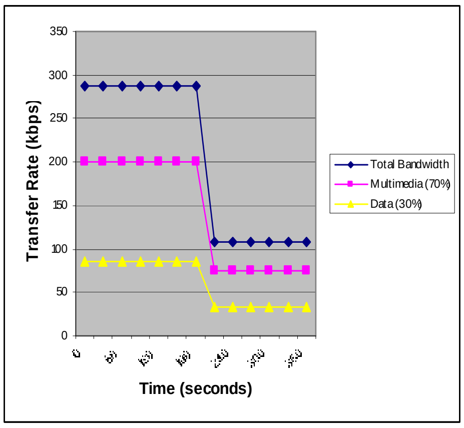

Fig.9. Bio-Inspired Scenario

This is because of the negotiations between node A and B, to allow a certain level of revenue to be maintained by prioritising the multimedia over the data traffic.

Based on this model, we can observe the totals of delivered traffic. As is visible in Fig. 10, it is evident that the gradient on the Conventional Packet Count is not as steep as the gradient on the Bio-Inspired Packet Count. This indicates better performance by the Bio-Inspired method even after the link failure marked at time 210 on the graphs.

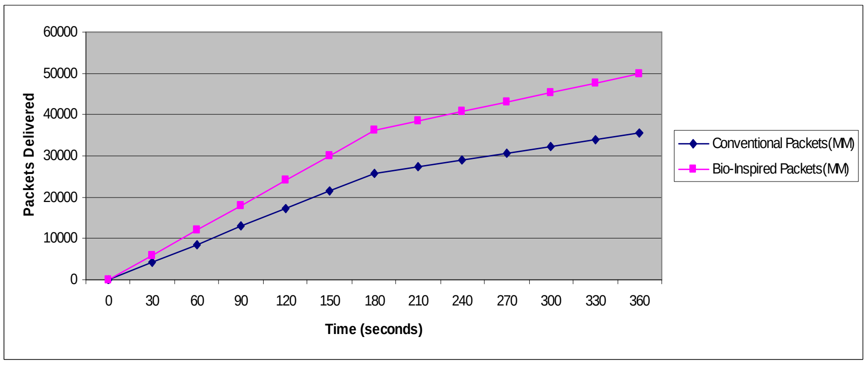

Fig. 10. Conventional Packets (Cumulative) vs. Bio-Inspired Packets (Cumulative)

5 Related Work

The authors of [6] distinguish management approaches by the *organizational model*, structured into centralized, weakly distributed and strongly distributed approaches. This model is helpful in a coarse categorization of paradigms in network management and to distinguish between traditional and more comprehensive approaches. The general trend is for network management to evolve towards strongly distributed paradigms and to provide more automation of management. However, only a few architectural and project-related solutions exist that provide a general and comprehensive approach to autonomic network management.

The FOCALE system presented in [3, 12] is characterized by a high level of autonomy, in that human interaction is only foreseen in the definition of business goals. However, the system is very complex and therefore difficult to understand in case of unforeseen failure of the management system itself. The same weakness affects the ASA architecture [13]. Although it is a generic architecture, which encompasses different abstraction layers and heterogeneous resources, it is characterized by a high level of complexity. The goal of an INM solution is rather the design of an autonomous system which keeps simplicity and flexibility to the fore, and provides a balanced level of autonomy with abstract interfaces..

Other network management architectures have been proposed, which focus on specific network environments (e.g. MANNA [7] in WSNs, Madeira [14] in P2P). However, their impact is limited to their target environment and their lack of generality doesn't address the requirements of a heterogeneous environment.

The *Autonomic Network Architecture (ANA)* project described in [5], [4], [2] has looked at developing a novel network architecture beyond legacy Internet technology that can demonstrate the feasibility and properties of autonomic networking. The problem field addressed by ANA is somehow close to the topics addressed in the INM model of 4WARD: they both aim at increasing the level of automation into the network and they follow a clean slate approach. Nevertheless ANA should be regarded as a generic architecture for autonomic devices, while INM leverages on a tight coupling of management functions with the services deployed on a device, like virtualization of resources or generic paths.

6 Conclusion

This paper proposes a new paradigm for the integration of management functions with its network components. The INM concept focuses on the embedding of management capabilities in all nodes and the potential which can be achieved with these management capabilities collaborating with each other. The current state of the INM framework is introduced. The framework must satisfy a diverse number for requirements but it will become a key component in the realisation of INM. A case study which looks at applying bio-inspired algorithms as low level routing techniques is investigated. This case study highlights the immense potential of the embedding of management capabilities at a low level in the nodes. Deployment of these management algorithms is shown with respect to the proposed framework.

Acknowledgments

The work described in this paper was partly funded by the European Union through the 4WARD project in the 7th Framework Programme Call 1.

References

1. A. M. Turing, The Chemical basis of morphogenesis. Philosophical Transaction of the Royal Society 237:37-72, 1952
2. Autonomic Network Architecture (ANA). Project funded by the European Union Information Society Technologies Framework Programme 6 (EU IST FP6). http://www.ana-project.org/
3. Brendan Jennings, Sven van der Meer, Sasitharan Balasubramaniam, Dmitri Botvich, Mícheál Ó Foghlú and William Donnelly: *Towards Autonomic Management of Communications Networks.* IEEE Communications Magazine, **45**(10):112-121, October 2007.
4. Christophe Jelger, Christian Tschudin, Stefan Schmid and Guy Leduc: *Basic Abstractions for an Autonomic Network Architecture.* In Proceedings of the 1st IEEE WoWMoM Workshop on Autonomic and Opportunistic Communications (AOC'07), Helsinki, Finland, June 2007
5. Fabrizio Sestinim: Situated and Autonomic Communication – an EC FET European Initiative. ACM SIGCOMM Communication Review, **36**(2):17-20, April 2006
6. Jean-Philippe Martin-Flatin, Simon Znaty and Jean-Pierre Hubaux: *A Survey of Distributed Enterprise Network and Systems Management Paradigms.* Journal of Network and Systems Management, **7**(1):9-26, 1999. Springer
7. Linnyer Beatrys Ruiz, José Marcos Nogueira and Antonio A. F. Loureiro: *MANNA: A Management Architecture for Wireless Sensor Networks.* IEEE Communications Magazine, **41**(2):116-125, February 2003
8. M. B. Miller, B. L. Bassler, "Quorum sensing in Bacteria“, Annual Review in MicroBiology, 2001, pp. 165-199.
9. N. Agoulmine, S. Balasubramaniam, D. Botvich, J.Strassner, E.Lehtihet, W.Donnelly, *Challenges for Autonomic Network Management*, accepted for 1st Conf. on Modelling Autonomic Communication Environment (MACE), Dublin, Ireland, October 2006

10. S. Balasubramaniam, D. Botvich, T. Gu, W. Donnelly, *Chemotaxis and Quorum Sensing inspired Interaction supporting Social Networking*, in Proceedings of 65th IEEE Vehicular Technology Conference (IEEE VTC Spring), Dublin, Ireland, April 2007
11. S. Balasubramaniam, D. Botvich, W. Donnelly, M. Ó Foghlú, J. Strassner, *Bio-inspired Framework for Autonomic Communication Systems*, (Editors Falko Dressler and Iacopo Carreras) in "Advances in Biologically Inspired Information Systems: Models, Methods, and Tools", Studies in Computational Intelligence, Springer Verlag, 2007
12.. Sven van der Meer, Willie Donnelly, John Strassner, Brendan Jennings and Mícheál O Foghlú: *Emerging Principles of Autonomic Network Management.* 1st IEEE Int'l Workshop on Modelling Autonomic Communications Environments, Dublin, Ireland, October 2006.
13. Yu Cheng, Ramy Farha, Myung Sup Kom, Alberto-Leon-Garcia and James Won-Ki Hong: *A Generic Architecture for Autonomic Service and Network Management.* Computer Communications, **29**(18):3691-3709, November 2006.
14. Madeira. http://www.celtic-madeira.org/index.html
15. http://cleanslate.stanford.edu/
16. http://www.ana-project.org/
17. http://www.atheros.com/