

Self-Forming Network Management Topologies in the Madeira Management System

Liam Fallon¹, Daryl Parker¹, Martin Zach², Markus Leitner², and Sandra Collins¹

¹ Ericsson Ireland Research, Ireland

{liam.fallon, sandra.collins, daryl.parker}@ericsson.com

² Siemens AG, Austria

{martin.zach, markus.leitner}@siemens.com

Abstract. This paper describes a novel approach to network management topologies where multiple customized topologies are self-configured, self-optimized, and maintained automatically by the underlying network of elements. An implementation of these self-forming management topologies as developed in the Celtic European research project Madeira is described. The self-forming topologies use peer-to-peer communication facilities provided by the Madeira platform running on each network element and give a view of the complete network topology, with customization optimised for individual management functionality. Finally, experiences in utilising these topologies are described, highlighting the benefits of this novel approach.

Keywords: network management topology, self-formation, self-configuration

1 Introduction

Most network management systems (NMS) in service today use layered management topologies that are a variant of the model proposed in the ITU standards [1] [2] originating from the late 80's. The network management topology in this model is a representation of the interconnectivity between the network elements (NEs) and the management systems, and can be characterized as centralized and hierarchical. This hierarchy introduces different levels of abstraction – from element management of individual NEs up to business management at the top of the Telecommunication Management Network (TMN) pyramid [1].

Management functions in each element manager read data from NEs connected to them to build internal logical topology mappings. Likewise, management functions in each network manager read data from element managers connected to them to build a logical topology mapping. Thus management information flow in both directions – up and down this static hierarchy – is cascaded, with information mapping performed at each layer. One consequence is, from a functional point of view, the five FCAPS disciplines remain rather separated. Interactions between these disciplines typically happen at higher management layers, or even through a human operator.

There are a number of drawbacks to this approach. There is a single network management topology in the network, it is static in nature: it is a representation of the

connectivity of the management system for the network. This topology is not necessarily optimal for every management function. It becomes very difficult to build advanced management functions that work across managers; as each manager is only aware of its local topological information and that of its subordinate NEs. As a consequence, this architecture inhibits managers at the same level from sharing topological information. This is a critical issue for multi-vendor, multi-technology management systems. Compounding the problem is the scale, transience, and diversity of elements in evolving networks. The task of keeping topology mappings consistent in dynamic networks is increasingly difficult; with more development effort required to keep track of what is managed rather than being focused on improving the management of the services the network provides.

Motivated by these issues we present a novel implementation of self-configuring, self-optimizing management topologies, based on a peer-to-peer (P2P) approach to network management, developed and implemented in the CELTIC research project Madeira [3]. The remainder of this paper is structured as follows: chapter 2 briefly summarizes related work; chapter 3 presents an overview of the self-organizing cluster topology mechanisms of the Madeira solution while chapter 4 provides details of the implementation. Chapter 5 describes our experiences of using such topologies in a test network and simulation environment. Chapter 6 concludes the paper and outlines future work.

2 Related Work

Networks which are predominantly static in nature are inherently straightforward to manage using existing TMN principles. In contrast, the networks we address in this paper have highly dynamic topologies. The target network we selected as our test-bed for the Celtic project Madeira is a large-scale wireless Ad-hoc network.

A centralised approach typically results in a high message overhead introducing a large waste of bandwidth; a critical resource in wireless environments. There are two potential approaches available to a central manager; polling or asynchronous notifications. Either approach has been seen to be inefficient, particularly for polling [4]; both approaches result in the central manager being a single point of failure, and bottleneck in the network. Therefore, this approach is not considered appropriate for large-scale wireless environments.

There exists however a number of management architectures currently used for Ad Hoc networks, based on a combination of both distributed and hierarchical approaches. The most notable of these systems can be found in [4], [5] and [6]. These proposals are largely based on specific clustering techniques: [4], [7] and [8] respectively. The superpeer management service described in [9] uses proximity based message broadcasting and NE capabilities to find and elect superpeers, building a single level topology that is optimized for latency. A taxonomy of various distributed management paradigms has been compiled in [10]. With respect to their taxonomy, our Madeira approach might be classified as following the strongly distributed hierarchical paradigm, with the additional advantage of using a completely dynamic hierarchy. The use of a dynamic hierarchy allows the Madeira system to

adapt to network failures and state changes, that [11] argues is essential to effective management of future large scale dynamic networks which must exhibit adaptive, decentralized control behaviours.

3 Madeira Management Topologies in Overlay Networks

An overlay network [12][13] is a logical network built on top of another network, with overlay nodes connected by virtual or logical links, each corresponding to a path, perhaps through many physical links, in the underlying network. The Madeira platform [3] is based upon the concept of Adaptive Management Components (AMCs); containers on NEs that run management software entities. AMCs provide services to those entities so that they can manage the NEs on which they are running and communicate with entities running on other NEs. This distributed management system has many advantages including low bandwidth usage, low message consumption, and scalability for large, dynamic networks.

It is evident however, that such an approach conflicts with some assumptions from a classical NMS approach. Traditional architectures assume a network with a relatively static hierarchical topological structure, in which:

- Aggregation and delegation occurs only as data moves up and down the hierarchy.
- Connections in the management hierarchy are controlled; adding and removing entities to a management structure is assumed to be a static task.
- The absence of a management entity is usually assumed to be an error.

In contrast, in the Madeira management overlay, element management and parts of network management run on NEs in the network itself in a flat peer-to-peer manner. Information is exchanged on east-west interfaces, allowing the distributed management system to understand its local environment and carry out environment-specific management tasks autonomically. Furthermore NEs can appear and disappear in normal operation, behaviour that conventional network management systems do not expect and has difficulty managing. The Madeira platform accommodates such behaviour through the use of management clusters which self-organise into a logical hierarchical structure, which can interact with management systems built using traditional management approaches. In this way the Madeira platform provides the base for building distributed and cooperative network management applications. The remainder of this chapter gives an overview of the functional aspects of the Madeira management clusters, with implementation details covered in chapter 4.

3.1 Topologies of Clusters and Cluster Heads

Management clusters are distributed structures that are formed in an ad-hoc manner by a set of co-operating NEs for the purpose of managing some network feature or function. Fig. 1 shows a network with its NEs formed into management clusters for two hypothetical management functions X and Y. These management clusters self-organize into a cluster hierarchy made up of cluster members and cluster heads. Each

cluster hierarchy constitutes a topology and many such cluster topologies may coexist, customised for one or more management functions.

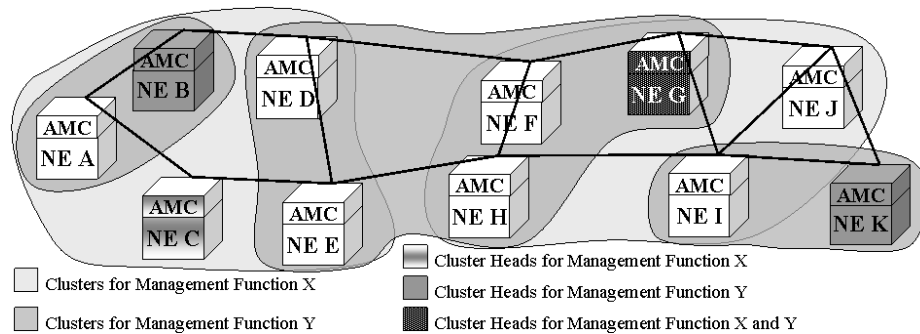


Fig. 1. Management Clusters for Management Functions X and Y.

The cluster head (or super peer) is the NE within a cluster that is elected to coordinate and publish the topology of that cluster. A cluster head may also aggregate and correlate data for a cluster or act as a mediation point. The cluster topology of a management function is tuned for that particular management function. A topology used by a fault management application is optimized for root cause detection of common faults on low level cluster heads, thus reducing the resources needed in higher level cluster heads. Cluster heads that correlate alarms are chosen because they have high processing power and reasonable memory availability.

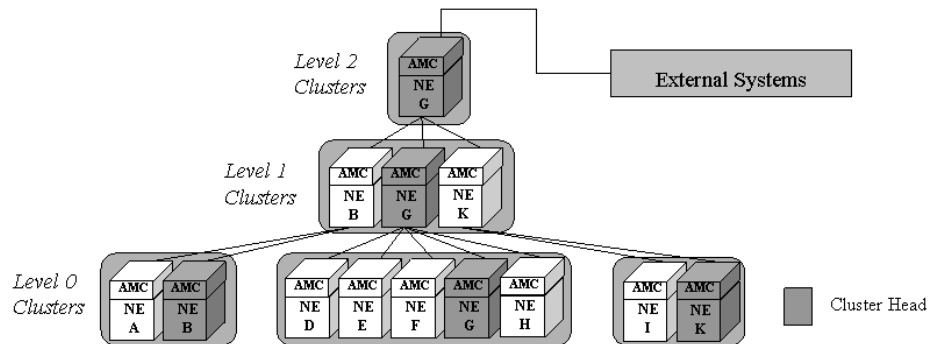


Fig. 2. Management Cluster Hierarchy for a Single Management Function.

Fig. 2 shows an example of a cluster hierarchy for a single management function. An NE in a level n cluster is the cluster head from a level $n-1$ cluster. NEs in a level n cluster elect a cluster head which represents the level n cluster in the level $n+1$ cluster. The hierarchy is self-configuring and re-forms dynamically and seamlessly whenever NEs are added, moved, or removed. New clusters may appear, merge, split, or disappear within a topology. Promotion and demotion of cluster heads may occur. New levels may appear or levels may disappear in the hierarchy of a topology. The topology on a single NE, a sub-tree of a topology or an entire topology can also be manually reset by a network operator or an external system if required.

The mechanisms for the formation of clusters, election of cluster heads, and supervising the levels in a hierarchy are controlled via parameters¹. A topology may re-form if any of the parameters that influence it are changed during network operation, causing the structure to self-optimize based on the prevailing network conditions. Parameters can be altered dynamically by direct operator intervention, automatically using policies [14], or as a result of changes in the network state.

Users of a topology such as an external system may register for topological events. Notification subscriptions may vary in scope from, an entire topology, a sub-tree of a topology, or a single NE. Examples of predefined events include:

- a NE being added to or removed from a cluster
- a NE is promoted to or demoted from being a cluster head
- a NE is promoted to or demoted from being the top NE in a topology

A management function may issue events specific to its function; for example a FM application may report correlated faults as events.

4 An Algorithm for Self-Forming Topologies

In this chapter, we introduce the main components and use case interactions in the Madeira implementation of self-forming topologies.

4.1 Topology Entities

The building block for all topologies is a Topology Entity (TE). A TE runs in the AMC of each NE for every topology that the NE is a part of. The TE is responsible for building and maintaining a local view of its topology for the NE on which it runs. The TEs for a particular topology communicate and co-operate to build and maintain the entire topology for the network as shown in Fig. 3.

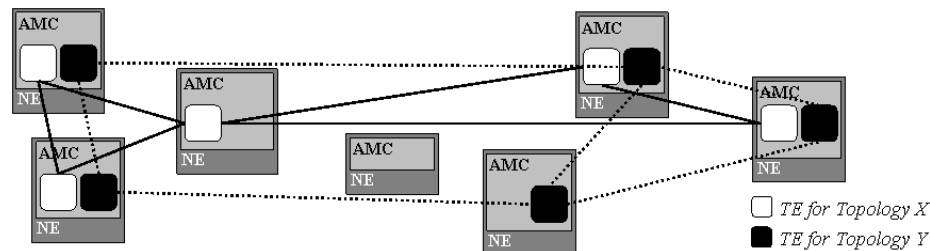


Fig. 3. Topology Entities for Two Management Functions Running in AMCs.

All TE software components are identical. A full topology is an aggregation of all local topology information for the TEs in that topology. A key characteristic to note is that when a topology is fully formed, it exists logically as a hierarchical layered

¹ Parameters might include type, technology, NE resource or service capabilities, NE and neighbour location, availability, accessibility, cost of use, topology structure and NE membership in other topologies.

structure, even though it is instantiated in a flat management overlay. A TE manages the structure of a topology for its local NE, keeping track of relevant data for the NE itself and the logical connections the NE has with TEs in other NEs.

If a TE appears at a certain level in a topology, it must also appear at every level below that level. A TE has either the role of cluster head or member at a particular level. A TE must be a cluster member at the highest level that it exists at in the hierarchy and a cluster head at every other level. A TE records its role at each level in a table. If a TE has a cluster member role at a level, the TE records the address of its cluster head and a flag indicating if that TE is the top cluster head. If a TE is a cluster head at a level, it records the addresses of each of the cluster members. Fig. 4 shows the structure of the TE tables for the topology shown in Fig. 2.

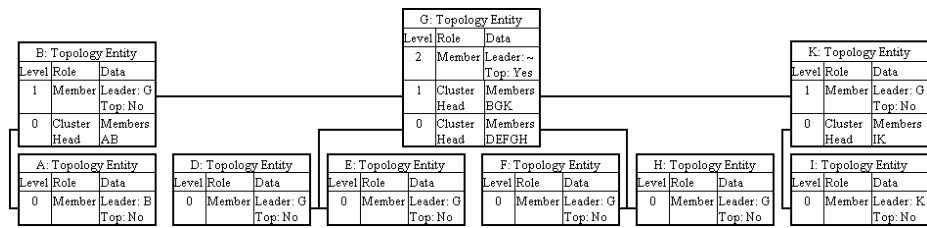


Fig. 4. Topology Entity Tables for a Topology.

4.2 Role Controllers

Tes communicate with each other to build and maintain topology structures, using role controllers to keep track of the topology at each level. Therefore, a TE has a Head Role Controller (HRC) for each level at which it is a cluster head and a single Member Role Controller (MRC) for its highest topology level. The HRCs and MRCs at a particular level in the topology co-operate with each other to maintain clusters and handle the topology at that level for the entire topology or for sub-trees of the topology. Fig. 5 shows the HRC and MRC co-operation for the topology of Fig. 2.

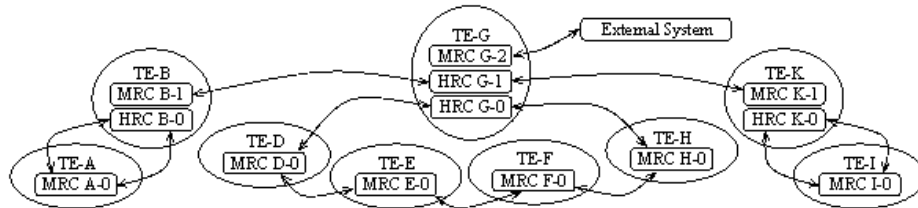


Fig. 5. HRCs and MRCs Co-Operating at Topology Levels.

4.3 Cluster Formation

In the following sections we present the TE lifecycle and describe various interactions for building, maintaining, and using network management cluster topologies.

destroying the MRC at that topology level, starting a HRC at that topology level, and starting a MRC for the next topology level. The cluster head query is passed to the HRC for handling.

When the HRC starts, it initializes a list to hold the addresses of its cluster members. Initially, this list includes just the address of the TE itself. It issues a cluster head promoted event to its subscribers.

Cluster Member Addition. Fig. 7 a) shows the message sequence used to add members to a cluster. A cluster head query may be handled by any HRC that is running or is started at the topology level specified in the query. If after checking its parameters the HRC decides to admit the new member, it reserves a place for the member and replies with a cluster head query reply. On receipt of the reply, the MRC checks its parameters to see if it is still allowed to become a member of the cluster of that HRC and if yes, sends a member confirmation message to the HRC. A MRC may receive more than one reply to a cluster head query. It always responds to and joins the cluster of the first HRC that replies and is acceptable, ignoring all other replies. The cluster member supervision on those HRCs removes the MRC reservation after a certain timeout has expired.

When the HRC receives a cluster member confirmation, it confirms the MRC as a member of its cluster and issues a cluster member added event to its subscribers. If the MRC had previously assumed it was at top of the topology it clears its top of topology flag and issues a not top of topology event to its subscribers.

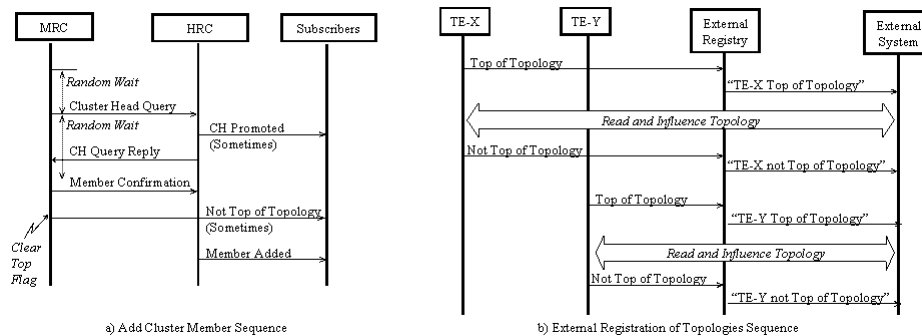


Fig. 7. Adding Cluster Members and External Registration of Topologies.

Cluster Member and Cluster Head Supervision. Once the cluster head and cluster member relationships have been established a supervision process is initiated to monitor these relationships. A cluster head periodically polls its members to ensure that they still exist. When a MRC is polled, its HRC expects a reply to its poll within a certain time. If the MRC does not reply within the time period, the HRC removes the MRC from its cluster. Once a cluster head has completed its member scan, if the HRC itself is the only member of the cluster, the TE demotes the HRC to a MRC.

Similarly the MRC expects to be polled by its cluster head periodically; if it is not polled by its cluster head for a set time period, the MRC assumes its cluster head has disappeared, and the MRC restarts itself.

Cluster Head and Cluster Member Orderly Removal and Shutdown. If a cluster member shuts down or restarts, its MRC sends a leave cluster message to its HRC if it has a cluster head. If the MRC had assumed it was the cluster member at the top of a topology and had set the “top” flag, that flag is cleared and MRC issues a not top of topology event to its subscribers. If the MRC is not polled for a set time period, or it gets an error when communicating with its cluster head, it assumes its cluster head has disappeared. In all these cases, the MRC restarts itself.

If a HRC cannot communicate with a MRC; when a HRC to MRC poll fails; or when a HRC itself is shut down, the HRC removes the MRC from its member list and issues a cluster member removed event to its subscribers. Similarly when a cluster head is shut down or restarted, it sends a cluster head shutdown message to all its cluster members and issues a cluster head demoted event to its subscribers.

Registration of Topologies and Communication with External Systems. When a cluster head realises that it is the top cluster head in a topology, it registers itself using a Top of Topology Event with an external registry such as a UDDI repository as being the entity that is publishing information externally for its topological hierarchy. An external system can subscribe to registry get the address of the top TE and can then open a direct communication session with that TE as shown on Fig. 7 b).

External systems may be informed that an address they are using is no longer valid, if the registry receives the top and not top events, ensuring connectivity to external systems even in case of unplanned outages of the top level TE.

5 Using Self-Forming Topologies in Management Applications

In order to validate our approach, we have built two distributed management applications using self forming topologies; an inventory retrieval application and a fault management application [15]. The validity of these applications for management of a real network has been shown on our test network, and we are currently performing a scalability analysis in a large-scale simulated environment.

5.1 Madeira Test Network

Our test network consists of 12 identical NEs² wirelessly connected using OLSR [13]. The topmost cluster head (CH) runs Apache Tomcat [16]; applications use it and Apache Pubsub [17] to publish their interfaces. A test OSS on a PC external to the network communicates with the topmost cluster head using web services. In our tests we did not observe a significant performance difference in the algorithm for various network configurations for the NEs.

The parameters used to determine the topology are set statically or using policies³ as detailed on the table below:

² HP tc4200, 1.73GHz CPU, 512MB RAM, 802.11 B/G interface., and Ubuntu Linux OS.

³ Policies are implemented as in the FM application [14].

Table 1. Parameter Values used to Determine the Topology in the Madeira Test Network.

Parameter	Value	Explanation
MaxClusterSize	3	Maximum NEs per cluster
SecsHoldReservation	5	Time before CH cancels member reservation
SecsBetweenMemberPolls	25	CH polling interval for member polls
SecsWaitMemberReply	5	Time CH waits for a reply to a member poll
SecsMinChSearchDelay	5	Minimum time a member waits before CH search
SecsRangeChSearchDelay	10	Window after minimum wait before CH search starts
SecsWaitChPoll	45	Member wait interval before assuming CH lost
MemberQueriesBeforeTop	3	CH queries before member becomes top

The structure of the topology built by the algorithm is largely influenced by the start-up sequence of the NEs. When all the NEs are switched on simultaneously⁴, it is difficult to predict where the NEs will appear in the resultant cluster topology. If however the NEs are started in sequence, the behaviour of the algorithm generally exhibits the following pattern. The first NE started becomes a level 0 cluster member, while the second NE becomes a level 0 cluster head and a level 1 cluster member. As more NEs are switched on, they become level 0 cluster members until the maximum number of NEs allowed in the cluster is reached. The next NE to be switched on becomes a level 0 cluster member in a new cluster, and the following NE becomes a level 0 cluster head for the new cluster, and generally becomes the level 1 cluster head. The algorithm continues in this way to build up the entire topology.

The time for topology formation to reach stability is highly dependant on the timeout values, see Table 1. Shutting down or restarting a cluster head causes part of the topology to be reformed, often resulting in a new cluster head being elected. When the algorithm has completed topology formation, the topmost NE starts Apache Tomcat and registers its address with the external UDDI repository. From this point on, the test OSS can use the inventory and fault management applications.

The inventory application publishes inventory information for all NEs by delegating queries from the topmost cluster head down through the topology, so that each subservient cluster head builds and aggregates the topology for its own cluster.

Our fault management application [15] demonstrated that the distributed approach exhibits a number of benefits, including scalability and robustness, inherent consistency between CM and FM related information, and the fact that simple alarm correlation rules are sufficient even for complex network scenarios. Apart from functional tests - the FM application worked well for faults like NE and communication link outages - we have performed extensive performance tests, with alarm rates of up to 10 per second processed by cluster members and cluster heads.

5.2 Madeira Simulation

A simulation environment is being used to assess the scalability of the Madeira management approach in general and self-forming topologies in particular. The preliminary results for a typical NE shown on Fig. 8 demonstrate the scalability of the

⁴ A rather unrealistic scenario in a real network.

Madeira management approach. Simulations of up to 100 NEs are tested, and show that the approach is scalable; NE CPU and memory usage stabilise after an initial increase, and the bandwidth usage by a NE is relatively low and constant for the entire simulation. Results for CPU and memory usage for cluster heads and the cumulative bandwidth use of the algorithm for all NEs in the network will be published when the simulation work concludes.

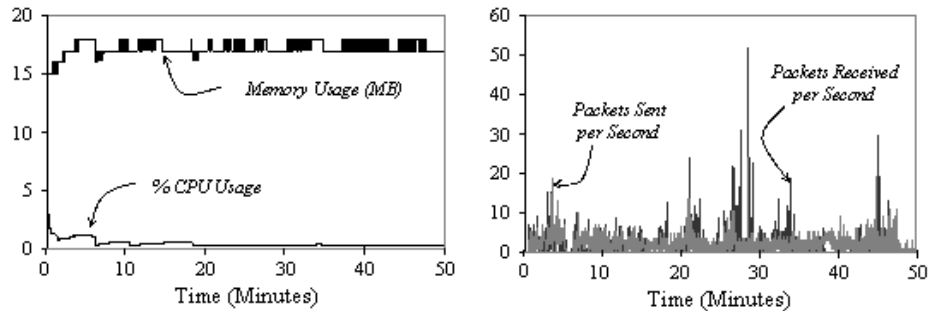


Fig. 8. Preliminary Results from Simulation Runs.

6 Conclusions and Outlook

In this paper, we have described the Madeira implementation of self-forming network management topologies. We have shown how logical, self-configuring, self-optimising, dynamic topologies can be deployed. Management systems no longer have to synchronise or manually configure topologies, and can delegate configuration and optimisation of topologies to NEs, while controlling their formation using parameters and policies. Management systems can monitor changes in topologies by registering for events rather than having to read and map changes themselves.

The use of management function-specific topologies means that topologies can be customised for particular management functions and can be self-optimised at run time to cater for changing conditions in the network, facilitating flexibility in the operator's management portfolio. Another advantage is that any NE can address other NEs in its topology cluster and so is aware of its environment, facilitating distributed autonomic management at local level. A cluster head can co-ordinate the management of that cluster, correlating data and acting as a mediation point, hence minimising delay, message overhead, and bandwidth consumption. This means that management applications can be built and run in a much more distributed manner than in a traditional management approach, facilitating autonomic management and reducing operator expenditure. In some cases, vendor-specific element and network management can be replaced by external systems that connect directly to distributed management functionality integrated on NEs.

We have also described the test scenarios carried out on the Madeira test-bed which demonstrate the validity of this distributed self-forming topology approach, in particular for two implemented applications; namely inventory and fault management.

The Madeira project is carrying out further research in the application of self-forming management topologies for distributed network management, and the scalability of the clustering approach is being evaluated using network simulators. Preliminary results show that the approach is scalable; the CPU and memory usage level out after an initial increase, and the bandwidth usage is relatively low and constant for the entire simulation.

References

1. ITU-T Recommendation M.3000, Overview of TMN Recommendations, ITU, Geneva, Switzerland (2000).
2. ITU-T Recommendation M.3010, Principles for a Telecommunications Management Network, ITU, Geneva, Switzerland (2000).
3. Arozarena, P., Frints, M., Fallon, L., Zach, M., Serrat, J., Nielsen, J., Fahy, C., Gorgalas, N.: Madeira: A peer-to-peer approach to network management. Wireless World Research Forum 16, Shanghai, China (2006).
4. Wenli Chen, Nitin Jain, Suresh Singh, "ANMP: Ad Hoc network management protocol", IEEE Journal on selected areas in communications, vol 17, no 8, August 1999.
5. C-C. Shen, C. Srisathapornphat, and C. Jaikaeo, "An Adaptive Management Architecture for Ad Hoc Networks", IEEE Communications Magazine, February 2003, pp. 108 – 115.
6. S. Sivavakeesar, G. Pavlou, C. Bohoris, and A. Liotta, "Effective Management Through Prediction-based Clustering for Next Generation Ad Hoc Networks", Proc. of ICC 2004, Paris, France, June 2004.
7. C. Jaikaeo, and C.-C. Shen, "Adaptive Backbone-Based Multicast for Ad Hoc Networks" Proc. of ICC 2002, New York, USA, April 2002.
8. S. Sivavakeesar, G. Pavlou, and A. Liotta, "Stable Clustering Through Mobility Prediction for Large-Scale Multihop Intelligent Ad Hoc Networks", Proc. of WCNC 2004, Atlanta, USA, March 2004.
9. G. P. Jesi, A. Montresor, and O. Babaoglu, "Proximity-Aware Superpeer Overlay Topologies", Proc. of Selfman 2006, Dublin, Ireland, June 2006
10. J. Martin-Flatin, S. Znaty, and J. Hubaux, "A Survey of Distributed Enterprise Network and Systems Management Paradigms," Journal of Networks and Systems Management, 7(1), pp.9-26, 1999
11. C. Adam, K.S. Lim and R. Stadler: "Decentralizing Network Management", KTH Technical Report, December 2005.
12. D. Andersen, H. Balakrishnan, M. Kaashoek, and R. Morris. The Case for Resilient Overlay Networks. Proc. HotOS VIII, Schloss Elmau, Germany, May 2001.
13. RFC3626 - Optimized Link State Routing Protocol, (OLSR). October 2003.
14. R. Marin, J. Vivero, Hai Nguyen, Joan Serrat, P. Leitner, M. Zach, and C. Fahy,. A Distributed Policy Based Solution in a Fault Management Scenario, Proc. of GLOBECOM 2006, San Francisco, USA, November 2006
15. Markus Leitner, Philipp Leitner, Martin Zach, Sandra Collins, Claire Fahy: Fault Management based on peer-to-peer paradigms. Proc. of IM 2007, to appear.
16. The Apache Tomcat Servlet Engine, <http://tomcat.apache.org>
17. The Apache Subscribe Web Services Notification (WSN) Implementation, <http://ws.apache.org/subscribe>