



HAL
open science

Self-configuration for Radio Access Networks

Javier Baliosian, Huw Oliver, Ann Devitt, Francoise Sailhan, Epifanio Salamanca, Boris Danev, Gerard Parr

► **To cite this version:**

Javier Baliosian, Huw Oliver, Ann Devitt, Francoise Sailhan, Epifanio Salamanca, et al.. Self-configuration for Radio Access Networks. IEEE Workshop on Policies for Distributed Systems and Networks (POLICY), Oct 2006, c, France. hal-03031514

HAL Id: hal-03031514

<https://hal.science/hal-03031514>

Submitted on 2 Dec 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Self-configuration for Radio Access Networks

Javier Baliosian*, Huw Oliver*, Ann Devitt*, Françoise Sailhan*, Epifanio Salamanca*,
Boris Danev* and Gerard Parr†

*Network Management Research Centre, Ericsson Ireland
Athlone, Ireland

Email: {javier.baliosian,huw.oliver,francoise.sailhan,ann.devitt,epifanio.salamanca.cuadrado,boris.xa.danev}@ericsson.com

†University of Ulster, Coleraine Campus
Northern Ireland

Email: gp.parr@ulster.ac.uk

Abstract— The ongoing work presented in this paper is aimed at bringing self-configuration capabilities into next generation radio access networks. We present the main concepts and architecture of our prototype. We also introduce briefly a novel strategy for foreseeing the outcome of enforcing policies integrating behaviour discovery techniques and finite state calculus into the conflict detection and resolution process. The main objective of this approach is to avoid instability problems of a distributed rule-based system.

I. INTRODUCTION

Mobile communications systems used to be homogeneous and tightly coupled systems. Nowadays there is a proliferation of new and diverse radio access technologies, new devices with the ability to connect to many of those networks and popular new IP services. Thus, next generation mobile communications will become a heterogeneous and complex environment composed of a highly diverse set of hardware and software. These systems will be complex enough to render extremely expensive the job of installing, configuring, optimising, maintaining and merging them in a timely manner. It may be unfeasible for humans to make decisive responses to demands, failures or changes on time.

The vision being widely considered as the long term solution for dealing with that complexity is the creation of new kinds of devices with the capacity to manage themselves. Autonomous or self-managed devices are the aim of several research groups but in particular this is the main goal for policy-based network management (PBNM). From our point of view, PBNM is about deploying into devices the knowledge of experts, the strategic view of the businesses and the preferences of the users and let the device decide what to do in the face of a highly dynamic context.

The ongoing work, to which this paper is an overview, aims to be a realistic approach for bringing self-configuring capabilities to next generation Radio Access Networks (RAN). We introduce a novel combination of the knowledge extracted from the working system with the structures and processes for evaluating policies and resolving their conflicts. This technique will be presented in Section V. Next section will present the solution rationale, in sections III and IV we summarise network and node architectures and in Section VI we present some use cases to illustrate how the system works.

II. RATIONALE

Our aim is to distribute the management architecture avoiding management hierarchies, whenever possible, in order to maximise the self-management potential of future network elements. Therefore, we are developing a distributed RAN management system with no central management entity but with some specialised nodes and scalable grouping strategies. We are taking advantage of known research on self-managed networks such as ontology-based management, policy-based network management and behaviour modelling.

Figure 1 shows the general view of our solution: a network management system where re-configuration decisions are made in a distributed manner by the network nodes. The re-configuration actions, driven by event-condition-action (ECA) policies modelled as in [1], may be performed on themselves or solicited to other nodes. They are not enforced directly on the devices. Inspired by NESTOR architecture [2], the changes are tried and propagated on a distributed network model first and then enforced on the devices. In our case the network model is an ontology-based model containing configuration constraints, propagation rules and workflows as described previously in [3]. Working in this way implies dealing with the inconsistencies of a distributed network model and with the traffic overhead resulting from maintaining such high-level information.

The local decision-making based on ECA policies is supported by behaviour knowledge globally discovered from the working system and modelled by means of Bayesian Networks [4]. In this way it is possible to foresee in some extent the impact of a given decision to avoid unstable or unwanted behaviour.

III. NETWORK ARCHITECTURE

It is not realistic to assume that a unique organisational structure (centralised, hierarchical or fully distributed) can face all the requirements resulting from the diversity of generic tasks (e.g., event management, resource discovery and data dissemination) involved in network management. Thus, despite the fact that we are working towards a distributed management solution, we also rely on centralised and hierarchical organisational structures when needed. Then, only the tasks that (i) by nature require an underlying semi-distributed organisation, or

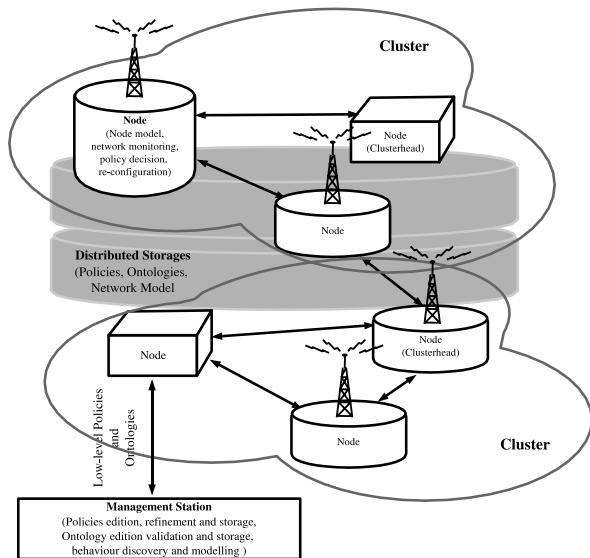


Fig. 1. Network Management Architecture

(ii) perform better under this structure (e.g., event aggregation and correlation), are achieved following a semi-distributed approach (§ III-B). In contrast, discovery of resources (e.g., policies, ontologies) follow a fully distributed approach (§ III-C). The centralised tasks are carried out by the *Management Station*, and the distributed tasks by the *nodes*.

A. Centralised Tasks

The *Management Station* is a special entity outside the managed network used to feed the system with semantic knowledge on the network model, policies and behaviour models. Despite the fact that it is a single logical entity, it may encompass several devices supporting the different modules. The main modules are:

Ontology Editor, Repository and Manager: The ontologies expressing the semantics and knowledge about the network configuration are edited, stored and deployed in the Management Station.

Policy Editor, Repository and Manager: The policies are edited and stored in the Management Station. As part of the policy lifecycle they will also be refined here into low-level, technology-specific policies, statically detectable conflicts are resolved as far as possible. Conflicts detected but not resolved in this process are communicated to the expert editing the policies.

B. Semi-distributed Organisation Management

As illustrated by Figure 1, our semi-distributed organisation structure is based on a backbone of nodes that are selected so as to perform routing, storage or control functionalities on local nodes (i.e., nodes in their vicinity) and therefore keep to a minimum the traffic induced by self-management tasks. These backbone nodes are also responsible for cooperating with each other to perform control and management of the overall network. More precisely, the backbone creation and maintenance

is dynamically carried out by a group management strategy that (i) includes a clustering algorithm that locally groups nodes under the responsibility of a particular node called cluster head or super peer, (ii) connects the cluster heads in a distributed and scalable way so as to form an overall backbone. Network management tasks are then executed on the top of a self-configurable group management service that deals with the network characteristics (e.g., dynamicity, network element connectivity). The identical strategy is used to deal with deployment of policies and ontologies and with the routing of events and remote actions.

C. Fully Distributed Resource Discovery

In order to support efficient resource discovery, several distinct overlays coexist, each being characterised by (i) various degree of expressiveness of the query language (i.e. structured, including logic specification, key-based), (ii) different underlying organisational structure, and (iii) different ways of routing queries. For instance, ontology and policy discovery is addressed with a DHT-like [5] solution suitable for location based on a single key. For location of resources (e.g. nodes), characterised by a set of attributes, the location solution consist in using a RDF-based query language to look for structured information, and a hypercup-overlay [6] (to propagate the queries) coupled with our aforementioned cluster-based solution (to disseminate events that maintain nodes aware of state changes).

IV. NODE ARCHITECTURE

Inside the node, the architecture designed (see Figure 2) combines techniques used until now for policy-based systems and self-managed devices plus some novel approaches for stability control based on behaviour discovery by means of data mining techniques and behaviour modelling using classical finite state machines (FSM). The combination of those machines with a Policy Decision Point which is also based on FSM produces a powerful and simple tool for foreseeing the consequences of enforcing a given policy. This allows us to work in a goal-based manner and to avoid undesirable effects of enforcing a policy such as non-stop cascades of changes or a ping-pong of changes between two or more nodes.

We are envisioning a new kind of configuration-wise autonomous network element with new software features in order to support its new self-management abilities (we call this a “Node”), but a Node may be more a set of functionalities than a single physical device. The structure of a Node has four main components: Network Element, Behaviour Model, Network Model and Event-Condition-Action Policy Decision Point (ECA PDP). A fifth important component is the Dynamic Conflict Resolution component described later in Section V.

a) Network Element: The Network Element in Figure 2 is the managed physical device as it is currently. It may offer complex management capabilities or a simple configuration interface. For compliance with legacy systems, the element in this architecture may be also the physical device plus an

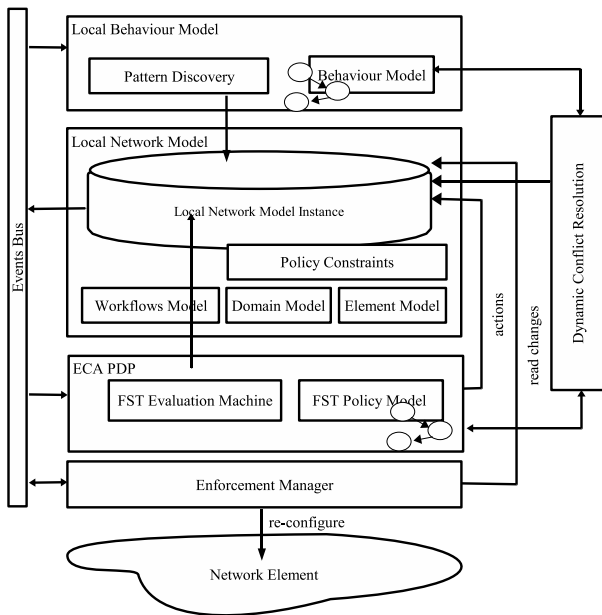


Fig. 2. Node Architecture

element management system with an appropriate north-bound interface.

b) Network Model: The Network Model, based on work presented in [3], is an ontology-based model of the node configuration and its relationships with other objects in the network. The Network Model is shared across all nodes in an Administrative Domain but the instances of the Network Model will differ from node to node. We refer to the sum of Network Models as a *Global Network Model*. Since the Network Model instances are not necessarily consistent, the Global Network Model instance may internally be inconsistent. Nodes will continuously interact to store and update shared entities such as links, VPNs, etc. by means of events and event subscription in order to converge their views without ever needing to achieve complete consistency.

c) Event-condition-action Policy Decision Point: This module is a classical PDP in charge of listening for events coming from the *Events Bus* and evaluating the conditions and the local policies for deciding the reconfiguration actions that need to be performed as a consequence. The events in the Events Bus may come from the network elements or from the different instances of the Network Model in each Node. The ECA PDP implementation is based on the TFFST-based model presented in [1] (TFFST stands for Finite State Transducers extended with Tautness Functions and Identities). This evaluation model is oriented to the resolution of policy conflicts and is intended to show good policy evaluation performance.

d) Behaviour Model: The Behaviour Model Component has a local instance on each node. The functionality of this instance is related to the function of that node within the management overlay. Nodes that are not part of the overlay backbone have a purely local behaviour model component

which aggregates and correlates events occurring within the node itself. This component summarises internal events for external presentation and identifies patterns of local events and their resulting states. This discovery functionality is based on an adaptation of the data mining techniques described in [7]. The behaviour model of nodes that are part of the overlay backbone and therefore have a view of their neighbourhood nodes includes events both internal to the backbone node and external to themselves. These external events are summarised information of local events on other nodes as well as actions or requests for actions in the network context. Again, the behaviour model has both a summarisation and a discovery function. Given the more global perspective of backbone nodes, the discovery function can identify patterns of local and global events representing the impact of node reconfigurations on the network context.

The patterns are modelled as probabilistic finite state automata (FSA) where probabilities are associated with the transitions between states.

V. BEHAVIOUR DISCOVERY AND CONFLICT RESOLUTION

As stated before, our system uses a TFFST-based model for obligation policies and their constraints and, at the same time, the behaviour discovery uses a FSA-based model as mentioned in Section IV-0.d. We are combining those two models using two complementary approaches with the aim of deciding between conflicting policies or constraining them using goal-based and stability criteria.

For dynamic policy conflict resolution, policy preferential weights are calculated on the fly on the basis of the tautness functions (TF) in the TFFST model. Patterns identified by the Behaviour Model which overlap with existing antecedents or conditions of event-condition-action policies impact on this resolution process by modifying the TFs with reference to the statistical probability of a given pattern of events. In this way, we reduce the priority of policies that have a high probability of resulting in undesirable events.

The second approach is aimed at minimising flip-flops and uncontrolled re-configuration cascades. The main idea is that the behaviour FSA model and the policy TFFST model can be analytically composed to derive predictions of the consequences of enforcing a given policy.

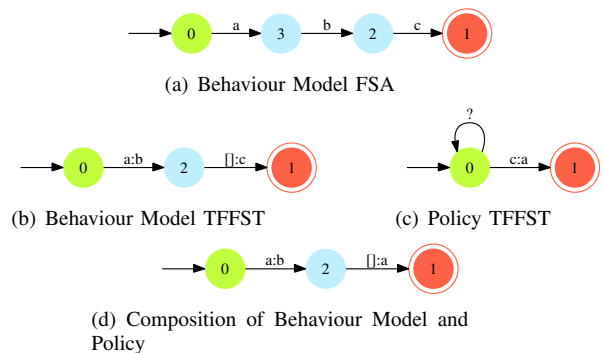


Fig. 3. Analytic Flip-Flop Discovery

As a high-level example, Figure 3(a) represents the discovered pattern of the action a followed by the action b and the event c . Figure 3(b) represents the same pattern as a TFFST (See [1] for more details). The TFFST in Figure 3(c) represents the rule “if c then a ” and Figure 3(d) is the composition of the transducers in 3(b) and 3(c). As in any binary relation, a composition of transducers is the creation of a new transducer such that the output of the first one is used as the input of the second one. In this new transducer we can see the action a in both sides of the transducer, the input and in the output. This means that performing action a eventually (or with a high probability) will cause the execution of the same action a again, a flip-flop behaviour that may be prevented by ignoring the rule modelled by Figure 3(c).

VI. USE CASES

To understand how the above architecture works, let us work through a few use cases. We will cover:

Bootstrap: How a node initially joins the network management platform and how the network configures its topology as a result of the new addition (Topology Discovery).

Load balance flip-flop: How the system adapts itself to an observed repetitive behaviour.

A. Use Case 1: Bootstrap

When a new node joins a network it knows only its set of capabilities and the roles it can play (for example an RBS, gateway, storage clusterhead, etc.). Its first action will be to obtain basic IP connectivity. It will use this to send out further requests. It will send out requests as events that will be picked up by its nearest clusterhead. The clusterhead will calculate where in the cluster the new node fits and assign a role to it. The new node will now query the network (using the location techniques mentioned in Section III) to locate and retrieve the related policies and ontologies (i.e. ontologies for specific services and the network model) from similar devices in the network or from the Management Station’s repository if there is no previous similar device. The node will generate further actions to register its interest in events relevant to it. The next request it will make will be for Topology Discovery. This is a request for information to populate its Network Model. It defines the relationship with other roles relevant to its role. This is internally constructed as an ontological model.

B. Use Case 2: Re-parent Flip-flop

When congestion occurs and is detected within a network, actions can be taken at differing timescales with different approaches. On a medium timescale, in a wireless access network, such as a WCDMA network, a typical action might be to alter which Radio Network Controllers (RNCs) control which Radio Base Stations (RBSs). Let us assume that a network element, an RNC, observes the local event that its load, over a given time period, is over a (statistical) threshold. The condition that triggers a virtual topology load balance is chosen (it is a better fit than the condition that triggers a real topology load balance) and the management action to

divert traffic to a different path is taken. This may result in the load threshold of a network element on the new path being exceeded and the same sequence of events occurs and the traffic is put back on the original path. This pattern would repeat indefinitely without behaviour modification. This behaviour is, in fact, observed in QoS-routed IP networks, called “Route Flapping” and is avoided by “Route Pinning”.

In our system this repetitive behaviour is detected by the Dynamic Conflict Resolution module as the appearance of the same event or action in both input and output of the graph representing the composition of the Behaviour Model’s FSA and the ECA-PDP’s TFFST and the policy triggering a virtual topology load balance is determined to cause this behaviour. The Behaviour Model now modifies the Tautness Function such that when this exact same event recurs the condition that leads to a real topology action will be chosen. Thus when the event occurs the RNC will now attempt to pass off one or more of its RBSs to an alternative parent RNC.

VII. CONCLUSION

This position paper presents the main concepts and architecture of a self-configuration for radio access networks prototype that we are currently developing. We also have given a glimpse of the power of integrating the behaviour discovered by data-mining techniques into the conflict detection and resolution process by introducing briefly a novel strategy to foresee the outcome of enforcing policies.

Despite the fact that at this time the prototype is in an early stage, the results obtained until now, together with the analytical work on behaviour modelling and policy evaluation integration seems promising.

We are following self-management and high-distribution paradigms with several complex issues to address such as rich data modelling, conflict resolution, consistency, location and traffic overhead. On the other hand, those paradigms promise a much more scalable and manageable solution for O&M in future heterogeneous radio access networks.

ACKNOWLEDGEMENT

The authors would like to thank Philomena Maloney for her help. J. Baliosian, F. Sailhan, and H. Oliver are funded by the European Community’s Marie Curie Host Fellowships.

REFERENCES

- [1] J. Baliosian and J. Serrat, “Finite state transducers for policy evaluation and conflict resolution,” in *Proc. of the IEEE POLICY*, 2004.
- [2] A. V. Konstantinou, Y. Yemini, and D. Florissi, “Towards self-configuring networks,” in *DARPA DANCE*, 2002.
- [3] D. Cleary and B. Danev, “Using ontologies to simplify wireless network configuration,” in *Proc. of the FOMI Workshop*, 2005.
- [4] D. Heckerman, “Bayesian networks for knowledge discovery,” in *Advances in Knowledge Discovery and Data Mining*. AAAI Press, 1996.
- [5] I. Stoica, R. Morris, D. Karger, and al., “Chord: A scalable peer-to-peer lookup service for internet applications,” in *Proceedings of the ACM SIGCOMM*, 2001.
- [6] M. Schlosser, M. Sintek, S. Decker, and al., “Hypercup–hypercubes, ontologies and efficient search on p2p networks,” in *Proceedings of the Workshop on Agents and P2P Computing*, 2003.
- [7] A. Devitt, J. Duffin, and R. Moloney, “Topographical proximity for mining network alarm data,” in *Proc. of SIGCOMM MineNet*, 2005.