Enabling Cognition in System of Systems: the Distributed Self-Growing Architecture

Marc Emmelmann and Bernd Bochow
Fraunhofer FOKUS
Kaiserin-Augusta-Allee 31; 10589 Berlin, Germany
Email: {emmelmann, bernd.bochow}@ieee.org

***Abstract*—This paper presents initial work towards an architecture suitable for purpose-driven, self-growing networking as realized by distributed cognitive decision engines within the network. Starting from describing basic modules enabling for self-growing, the paper derives a logical architecture for realizing the concept at various network levels. Mapping results to the UMTS stratum model shows potential for immediate applicability of the concept to deployed networks.**

I. Introduction

The novel concept of purpose-driven, self-growing networking [1], [2] addresses challenges of future wireless networks such as how to prove the efficiency and sustainability of co-existing networks, or how to handle the increased complexity in network operation and management arising with the variety of coexisting technologies and network components. Thereby, a self-growing network coexists, collaborates or integrates—potentially in symbiosis—with collocated networks utilizing their service or geographical extend to augment network capacity, or operational constrains such as energy consumption [5]. The self-growing process including network operation and management is realized by focused cognitive decision making controlling network and node reconfiguration. Depending on the ability of its network components, a self-growing network can autonomously and on demand switch between dedicated, generally pre-defined purposes [3].

For example, the potential of incident-triggered purpose changes can be illustrated by the most recent tsunami crisis in Japan: remaining telecommunication infrastructure broke partially down as the network tried to offer optimum voice services which conflicted with the tremendous amount of users attempting to place a call. Doing the aftermath, operators indicate that changing on-demand the purpose of the network from 'providing optimal quality' to 'providing maximum number of calls' (at the lowest acceptable quality) could have avoided this breakdown. An additional desirable purpose change would have been to further switch into an 'energy optimized operation' after having accommodated the bulk of initial emergency calls wherein service is only provided on a time-limited base.

The research leading to these results has received funding from the European Community's Seventh Framework Program (FP7/2007-2013) under grant agreement CONSERN no 257542.

This paper describes the first step towards developing a self-growing architecture enabling cognition in system of system. Its unique contributions focus on the description of the functional architecture enabling the novel self-growing paradigm. In particular, Section II introduces two types of basic modules enabling self-growing and discusses which functions have to be provided by each module in order to realize the self-growing paradigm. Afterwards, Section III composes those basic modules into a logical self-growing architecture. It therefore presents a high level view to the layered cognitive control architecture in order to elaborate on the basic interfaces encountered, i.e., the reference points of the architecture. Next, briefly mapping the architectural model to the 3GPP UMTS stratum model [4] emphasizes that self-growing can an easily applied throughout suitable evolution steps of future networks. The outlook on upcoming work given in Section IV concludes the paper.

II. Realisation of Self-Growing Enablers

A. Modules and Functions enabling Self-Growing

Enabling self-growing involves in general two classes of modules: a cognitive engine (CE) and a class of modules implementing a certain function (F) offered to the network. The CE realizes the cognitive decision capability for self-growing networking whereas the F interacts with the environment. This interaction is (partially) controlled by the CE receiving context information from the F and responding with coarse, rather unspecific ranges of operation (e.g. setting ranges of permissible frequency bands to operate in, or allowing a local decision at the F for setting its transmission powers as long as it is below a certain threshold), up to providing stringent, mandatory rules to be obliged by the F (e.g., set transmission power to a given, fixed level, or enabling/disabling certain interfaces for communication with a given other node).

1) Function (F): Immediately interacting with the environment, a F may provide functionality to enable IP connectivity between two networks, or as a sensor providing temporal- and special-specific context information. As a module enabling self-growing, the F may provide the following functionality: a) context provisioning and aggregation, b) communication service provisioning, c) knowledge provisioning, d) control & configuration services, or e) decision making & learning if

enhanced by additional cognitive capacity. The actual parameters and responses, or even the availability of certain functions depend on the capabilities of the device instantiating those functions of the function module. Hence, the only mandatory functions per F are:

- discovery of suitable CEs within the F's vicinity,
- message exchange between F and CE, and
- reporting a list of capabilities of the F.

Due to the evolution of networks as seen by the self-growing approach, the used description format for the capability exchange shall specifically support the evolution of schemas over time without requiring all the data consumers to be changed and should inherently facilitate data merging even if the underlying schemas differ.

2) Cognitive Engine (CE): Cognitive engines for self-growing may enable synergies among different network providers, within one operator's domain among nodes, and directly between nodes without the operator's control by having cognitive capabilities on individual devices. Correspondingly, we find CEs at the administrative, at the operational, and at the node (functional) level. This adds another degree of freedom in designing an architecture supporting self-growing: CEs may exist in one compact module (coping with the administrative, operational, and functional level) or may be distributed within the system having one CE per architectural level. Accordingly, functions of each sub-module have to be accessible by other sub-modules and each sub-module has to be capable of either directly or indirectly accessing functions within functional units. As a higher level CE's sub-module may indirectly accesses a F via lower level CE's sub-modules, functionality provided by CE's sub-modules has to encompass capabilities of Fs.

B. Architectural constrains for Self-Growing

In order to survive major changes in underlying network topologies, the architecture should avoid single points of failure, e.g. by providing redundancy or distributing CE functionality across several entities (on different network nodes). Primarily, we follow the latter approach as functionality of the CE can be grouped into cognition and control on a functional, operational, and administrative level. Hence, functions of a full self-growing node are under the control of (local) cognitive engine. Upon a node entering a network, its CE makes itself (and thereby implicitly the availability of the node under control of the cognitive engine) aware to the cognitive engine at operational level.

A sudden change in the underlying network topology may cause an interruption of the former communication path between cognitive engines at each level; also new nodes may join the network. Loss of communication as one indication of topology change is pair-wise detected by participating nodes using service discovery or service announcement features to re-establish hierarchical communication between cognitive engines. If a change in topology causes networks to merge, several cognitive engines at the operational or administrative level might be discoverable and announce their willingness to take cognitive control of the system. The detection of this duplication in available functionality by the service discovery scheme allows redundant engines to negotiate which one takes temporal control of the system and which engines should turn into a dormant state. Such detection can be enabled by a dedicated function specialized on providing communication between nodes and on providing service discovery or service announcement. In case of the network partition, CEs on the operational and/or administrative level might not initially exist in network fragments. Detecting the absence of such cognitive functionality allows appropriate actions to be taken: for example, CEs discovering missing cognitive functionality may cause an instantiation of a new CE on selected network nodes based on existing CE prototypes. Alternatively to instantiation, dormant cognitive engines can reactive themselves upon detecting the absence of functionality that they can provide.

Apart from service / capability discovery, interfaces between cognitive engines and between cognitive engines and functional units on network nodes should a) encompass a core set of functions via statically defined service primitives, b) allow for node / technology specific implementation of means to achieve actions requested by service primitives, and c) enable dynamically enabling or removing (core) functionality on nodes. This can be achieved by separating the initiation of a function call as defined by the interface between units in the architecture from the implementation of its functionality. Implementation of functionality can be stored in form of an implementation repository extendable by implementations of functionality not originally present in a particular entity. Means for dynamically adding (or removing) parts of the repository can be provided, e.g., in combination of an entity specialized on providing communication services between entities and an entity managing the node itself. Figure 1 illustrates the functional key components as well as how communication units form a virtual communication bus abstracting from potentially distributed implementations.

An open distributed object computing architecture building its functionality on top of OSI transport layer may be one suitable solution achieving these goals as it decouples the self-growing functionality of a network from the actual realization of underlying network services such as addressing and establishing transport streams between involved entities. Also, service discovery / capability announcement can be assumed to be available in realizations of distributed object computing.

III. Architecture Model

Following the enablers of self-growing discussed in Section II, the architecture model will encompass three layers of cognitive control:

The *functional layer* considers network nodes or whole networks as a collection of functions. Two reference points constitute on this layer: a) the CE-F interface between cognitive engine and the F which can be understood as a control and configuration interface for reconfigurable device and b) the CE-CE interface between cognitive engines. The latter

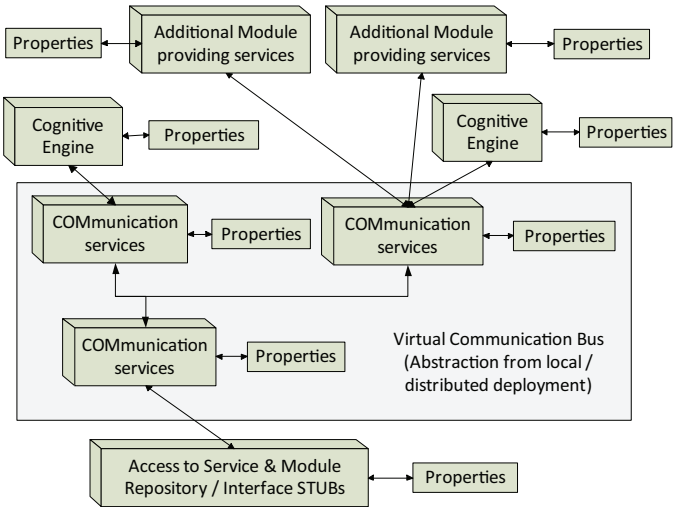

Fig. 1. Functional Key-Components of Self-Growing Architectures

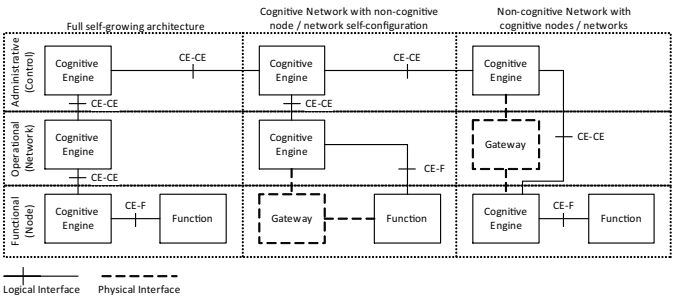

Fig. 2. Basic applications of a self-growing architecture to various topologies

can instantiate as both an intra- or inter-layer interface. The *operational layer* considers cognitive coordination of nodes or networks that may implement their own cognitive control capacity. On this layer the CE-CE reference point constitutes as an interface between distributed cognitive decision-making. Typically, the self-growing attribute is realized on this layer, in particular coordinating purposes and life cycles [3], i.e., node and network configuration, topology changes, coexistence and integration, etc. The *administrative layer* collects cognitive control capacities and interaction between cognitive engines required to coordinate between collections of nodes or networks each under their dedicated cognitive control. On this layer the CE-CE reference point mainly constitutes between cognitive engines of the operational layer. The interface thus may have a dedicated objective on the exchange of knowledge (e.g. context, rules or policies) enabling decisions on coordinated activities of self-growing networks e.g., incentive based collaboration or integration of self-growing networks. Clearly, more restrictive security requirements apply to interactions between cognitive engines on this layer.

In a practical implementation boundaries between layers may be fuzzy to some degree since certain topologies or configurations may require cognitive engines to coalesce across layers. Figure 2 includes the illustration of a 'cognitive network with non-cognitive node configuration' in which a coordinating cognitive engine also controls a node level functionality due to a lack of cognitive control features on this layer. In this case the realization of the reference point CE-F between cognitive engine and function is cross-layer and the CE-CE interface is not implemented in this direction.

From a perspective of potential migration paths towards a self-growing architecture, Figure 2 defines three basic configurations. A full self-growing architecture consists of reconfigurable nodes or networks in that nodes or networks associate with (potentially collocated) cognitive decision-making capacity, which in turn controls collocated or distributed functions (implementing reconfiguration capacity) via the CE-F interface. Distributed cognitive engines are communicating and coordinating utilizing the CE-CE interface to control the self-growing capacity of the compound system. In addition, a CE-CE interface is realized to enable coexistence, coordination and collaboration between self-growing systems, potentially exchanging information required to prepare and initiate merging of these self-growing systems into a single system.

A cognitive network consisting of nodes or networks without dedicated cognitive capacity consists of nodes or networks realizing a certain degree of configurability controlled, for example, by implementing a control and configuration interface per node or by some dedicated gateway entity responsible for configuring nodes and network (which in turn might be collocated to a node). This configuration capacity is assumed to be not of cognitive nature, i.e., algorithmic or heuristic. In this case a coordinating cognitive engine, mainly responsible for implementing the self-growing attribute, may also take responsibility for configuring nodes or networks accordingly. Hence, the operational layer constitutes both the CE-CE and the CE-F reference points to allow cognitive engines associated with this layer to communicate with each other (e.g. in a distributed realization) and with functions throughout the network that must be controlled and configured. Cognitive engines must rely on gateway functions either providing access to node or network configuration functions, representing this functionality as a proxy, or as a controller or service gateway with some additional abstraction of the functions controlled. As a proxy it may also represent the configuration capacity of a collection of nodes or networks. Next layer cognitive functions are considered equivalent to a full self-growing architecture as discussed earlier in this section. This architecture enables planning and decision-making on purposes and life cycles in it's own network comparable to a full self-growing architecture. Yet it is restricted in the set of life cycles and purposes it can attain by the configuration capacity reflected by the gateway, which might be significantly less than provided by single nodes, since individual policies may apply potentially prohibiting certain configurations.

This architectural model applies to a number of options based on existing network architectures. For example, a WSN consisting of low-profile nodes may support control of sensor acquisition rate, communication frequency and communication

routes via a gateway providing an API or a Web interface. Usually optimized for increased battery lifetime a coordinating cognitive engine can use these functions for reconfiguring the WSN to increase sensor acquisition rate and to reduce communication delay by route optimization at the same time. For the cost of increased battery drainage of some WSN nodes, collaborating nodes may gain knowledge required to attain the targeted purpose e.g., to determine the area impacted by an incident and providing communication services for this area.

A network consisting of cognitive nodes or networks without per-network cognitive coordination of self-growing capacity. This architecture consists of reconfigurable nodes or networks in that nodes or networks associate with (potentially collocated) cognitive decision-making capacity, which in turn controls collocated or distributed functions (implementing reconfiguration capacity) via the CE-F interface. Distributed cognitive engines are communicating and coordinating utilizing the CE-CE. Since this architecture lacks a cognitive control of the self-growing capacity across a collection of nodes and networks (on the operational level) it rather constitutes a loose collection of coexisting systems that can be coordinated in a self-growing manner (on the administrative level) but lack the inherent support (and knowledge) for attaining full self-growing capacity.

This architecture thus enables planning and decision-making on purposes and life cycles across networks but doesn't provide full functionality when coordinating its cognitive nodes / networks in a self-growing manner. That is, coordination across networks cannot make full use of the self-configurability of participating nodes / networks although these nodes are collaborative in their cognitive control. This is based on the assumption that a fully decentralized and collaborative decision making scheme a) cannot achieve the same or better performance than a partly centralized decision making architecture, and b) cannot control neighboring nodes / networks remotely that do not provide local cognitive control capacity.

This architectural model applies to a number of options migrating existing networks into self-growing networks. For example, deploying cognitive nodes into an existing network and deploying cognitive control to this network can be managed independently, increasing heterogeneity in the first place and increasing complexity of cognitive control on the operational layer. This evolution of cognitive control across layers towards a full self-growing architecture should avoid the need for synchronizing complex intermediate steps. Hence, a network could be partitioned (regarding its cognitive capacity) by creating dedicated sub-networks and collecting self-growing nodes / networks and functional (non-cognitive) nodes logically into separate domains. The coordination then is maintained on the administrative layer the same way as for heterogeneous networks

From the discussion above a stratum model can be developed resembling that given earlier for UMTS. The purpose of this model here is to depict the interaction (in terms of information exchange across interfaces) of cognitive engines

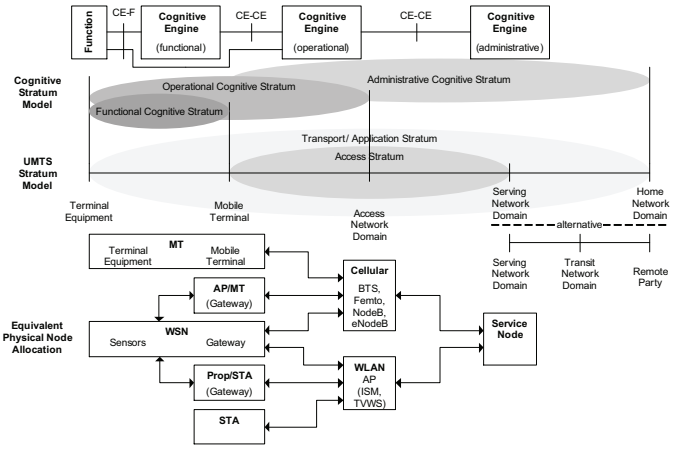

Fig. 3. Stratum Model of Cognitive Engines realizing Self-Growing

in this architecture. As shown in Figure 3 the role of network nodes in the architecture is conventional but slightly more complex due to their configurability and other self-x capacities. For the stratum model it is assumed that cognitive decision-making is incorporated either as a cognitive function with a network node (potentially one or more of those shown in Figure 3) or with a dedicated node hosting a decision engine e.g., as a dedicated service node.

IV. Conclusion

Based upon analyzing required functionality enabling self-growing in system of system, this paper has derived a logical architecture enabling cognition for self-growing at various 'network / node levels'. A stratum-model-based comparison with the UMTS has shown that self-growing functionality can even easily be added to deployed systems. An initial evaluation of the self-growing paradigm will be conducted within the CONSERN project [4] in form of a proof-of-concept prototype.

References

[1] N. Alonistioti, A. Merentitis, M. Stamatelatos, E. Schulz, C. Zhou, G. Koudouridis, B. Bochow, M. Schuster, P. Demeester, P. Ballon, S. Delaere, M. Mueck, C. Drewes, L. V. der Perre, J. Declerck, T. Lewis, and I. Chochliouros, "Towards self-adaptable, scalable, dependable and energy efficient networks: The self-growing concept," in *UBICOMM 2010*, 2010.

[2] B. Bochow, M. Schuster, L. Thiem, and J. Tiemann, "A novel system paradigm for self growing wireless networks," 2010. [Online]. Available: http://publica.fraunhofer.de/documents/N-132320.html

[3] B. Bochow and M. Emmelmann, "Purpose-driven, self-growing networks a framework for enabling cognition in systems of systems," in *GreeNET Workshop*, 2011.

[4] CONSERN, "COoperative aNd Self growing Energy awaRe Networks," July 2011. [Online]. Available: https://www.ict-consern.eu/

[5] E. De Poorter, B. Latre, I. Moerman, and P. Demeester, "Symbiotic networks: Towards a new level of cooperation between wireless networks," *Wireless Personal Communications*, vol. 45, no. 4, pp. 479–495, Jun. 2008. [Online]. Available: http://dx.doi.org/10.1007/s11277-008-9490-5