Towards a Framework for Self-Adaptive Reliable Network Services in Highly-Uncertain Environments

A. Ceccarelli[1], J. Grønbæk[2], L. Montecchi[1], H.P. Schwefel[2,3], A. Bondavalli[1]
1- University of Firenze, Viale Morgagni 65, I-50134, Firenze, Italy
{andrea.ceccarelli, lmontecchi, bondavalli}@unifi.it
2- Dept. of Electronic System, Aalborg University, Denmark, ljg@es.aau.dk
3- FTW, Wien, Austria, schwefel@ftw.at

***Abstract*—In future inhomogeneous, pervasive and highly dynamic networks, end-nodes may often only rely on unreliable and uncertain observations to diagnose hidden network states and decide upon possible remediation actions. Inherent challenges exists to identify good and timely decision strategies to improve resilience of end-node services. In this paper we present a framework, called ODDR (Observation, Diagnosis, Decision, Remediation), for improving resilience of network based services through integration of self-adaptive monitoring services, network diagnosis, decision actions, and finally execution (and monitoring) of remediation actions. We detail the motivations to the ODDR design, then we present its architecture, and finally we describe our current activities towards the realization and assessment of the framework services and the main results currently achieved.**

Keywords-pervasive systems; monitoring; ODDR; diagnosis; decision; measurement uncertainty; adaptive model generation

I. INTRODUCTION

Trends on future networking systems include convergence towards pervasiveness and heterogeneity, where a multitude of nodes and sensors interacts among themselves and with infrastructures to execute services, and where system boundaries are unknown [4]. In such scenarios, a user relies on a large variety of end-user services in infrastructure and ad-hoc networks; amongst them, resilient and real-time constraints are often mandatory requirements for services in order to support critical applications, where unreliable connectivity may lead to lost revenue, injury (e.g., safety-related issue), etc. Due to their highly dynamic nature, such systems require constant adaptation to changes in the environment.

The continuous evolution and the high decentralization of the networking architectures motivate an approach for an end-node to adopt online adaptive solutions for monitoring, diagnosis and remediation. This approach enables to make use of the diversity of different networks (technology, providers and operational characteristics) to provide improved resiliency and trust of the end-node network services without requiring network support.

The dynamicity of the system, the infrastructure and the environment calls for self-adaptive solutions for active and passive monitoring. For similar reasons, new diagnostic and decision methodologies, that rely on techniques for online model generation, are needed too. Such activities need to be performed almost completely automatically, to reduce the human intervention and the related costs.

The end-node driven fault management [5] approach may improve the ability to observe the network, diagnose its actual status and decide possible remediation actions, whenever needed. Several challenges arises in end-node driven fault management for resilient and real-time services in pervasive and heterogeneous environment, due to the increasing network dynamicity, unpredictability and heterogeneity. For example, i) the system may change quicker than a sufficient amount of training data can be obtained (training data reflects current system properties to a varying extent); ii) threat and fault definition and occurrence rate undergo a permanent update; and finally iii) the configuration and environment of systems and infrastructures are not defined a-priori and online adaptation is needed. In such context faults are rarely directly observable, consequently the end-node must rely on unreliable (noisy, ambiguous, missing, contradictory [22]) and uncertain observations based on existing network traffic or active probes to infer about network states. This may lead to significant imperfections of the network state estimate and executed remediation actions.

In this paper we focus on the joint observation, diagnosis and remediation of faults in networking systems through an end-node driven fault management approach to improve timeliness and resilience of services. We introduce the framework *ODDR* (*Observation, Diagnosis, Decision, Remediation*) for self-adaptive and online observation, diagnosing and consequent decision on possible remediation actions. Additionally, the execution of the remediation actions themselves and the monitoring of their outcomes are part of the ODDR functionalities. Therefore the ODDR acts as a middleware service that operates in an end-user device; it makes online decisions to manage network communication and to improve the resilience of the related user services.

We further describe our current work towards the development of the ODDR and the implementation of its main functionalities. In particular, we are currently investigating three different directions that address three different parts of the ODDR framework.

The first direction is an approach to improve the network diagnosis through confidence in network observations; this direction focuses on the observation functionalities of the ODDR and on their integration to the diagnosis ones. The second direction is the development of decision and remediation strategies that mitigate unreliable observations or diagnosis; this direction aims to improve the ODDR decision functionalities and integrate them with the monitoring and diagnostic functionalities. The third direction is related to the possible development of automatic and online adaptation of the decision mechanisms to match the continuous evolution of the system; this direction aims to implement and integrate the online and self-adaptive decision and remediation functionalities of the ODDR.

The rest of this paper is organized as follows. Section II shows the motivations of the ODDR and its novelty, Section III describes the ODDR architecture, Section IV deals with the use of measurement uncertainty to improve diagnosis, Section V addresses the problem of reliable decisions despite unreliable observations and diagnosis, and Section VI is devoted to adaptive online decision mechanisms. Finally, conclusions are in Section VII.

II. ODDR Motivations, Improvements to the State of the Art and Main Challenges

In this section the background and motivation of the ODDR framework are presented.

A. ODDR Sample Usage Scenarios and Related Benefits

In Fig. 1 a generic scenario for the usage of the ODDR is shown. The ODDR framework can be considered as a set of middleware layer services providing interfaces for both end-user and other middleware layer services, on top of a traditional IP protocol stack. The ODDR is expected to have access to observations on all layers in the protocol stack and may provide reconfigurations of these as well. An end-user service may ask for reliable communication services to the ODDR along with its end-user service requirements. The ODDR will subsequently attempt to manage the communication to avoid service failure.

We now define three specific examples to motivate the possible usage of ODDR as a decentralized approach to improve service dependability and timeliness from a decentralized end-node perspective. These examples include safety-critical issues; in fact the ODDR aims to represent a feasible alternative when traditional solutions for safety architectures are precluded due to inherent system complexities, and safety-critical issues are not avoidable.

Car2car and car2infrastructure networks (VANET, Vehicular ad-hoc networks [1]). Vehicles are expected to drive forward the use of wireless communications in mobile ad-hoc networks. Future generations of car electronics may provide information about locations for road pricing or alarm calls to emergency services using vehicle-to-infrastructure (e.g., via UMTS, GSM, WLAN) communication. In addition, inter-vehicle communication may allow exchange of information about road conditions or car control data to enable assisted driving services as platooning [2] or evasive maneuvering. For such services real-time and dependable

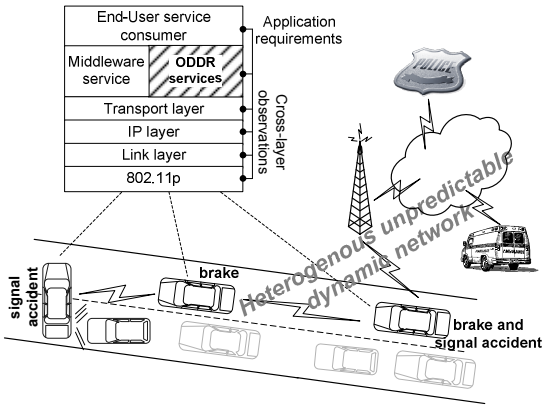

Figure 1. ODDR services: sample usage.

communication is crucial. The ODDR may attain several roles. An example is the real-time decision on the best communication path between cars: when a car is emergency-braking, it requires to notify its status to cars behind. The reliability and timeliness of the network communication is consequently mandatory: the ODDR module has the ability to diagnose the network state, and decide upon the best end-to-end path (e.g., through UMTS instead of the ad-hoc network) that expectedly is not influenced by the particular fault and provides the needed reliability and capacity.

Industrial use case: cable replacement problem. In many industrial applications cabled solutions are still being preferred to wireless ones, especially in closed networks, due to their consistent performance, security and reliability. Yet, wireless solutions promise to improve flexibility. The problem of maintaining a high reliability and efficiency in wireless communication solutions has been studied in [3]. The work considered software upload and data download for diagnostic purposes in relation to a Driver Machine Interface (DMI) onboard a train. Maintenance personnel may access the DMI via a mobile computer while the train is nearby or in a remote location. The end-user service may simply be an FTP transfer; to maintain efficiency, requirements are set to how long time the file transfer may take. The ODDR may decide upon the needed goodput while minimizing overhead from unnecessary remediation actions. It can benefit from service state knowledge e.g., that a file transfer is nearly complete, and decide not to react to a (potentially falsely) diagnosed fault to avoid jeopardizing the successful transfer by risking a timely and potentially unreliable network fail-over.

Health Care: Emergency Services. Wireless communications are also finding relevant applications in health care. An example is an on-board computer in ambulances which collects information about a patient during the treatment in the pre-hospital phase [8]. This information must be sent to doctors in the emergency room before the ambulance arrives enabling them to prepare the treatment and initiate it as soon as the patient arrives. During the emergency response and transfer of the patient information via UMTS, the connection may be lost or show degraded performance due to a fault in the provider network. Through exchanging information with other mobile devices in the proximity, and diagnosing the

current state of the network, the ODDR knows which local WLANs are most reliable and chooses to probe a few of these to assess their current state and maximize the probability they can convey the information before they get out of reach due to mobility.

B. State of the Art and Expected Improvements and Novelty

The interplay between monitoring and fault management components to correlate observations, diagnosis outcomes and decision/remediation actions has been largely explored in the past; for example several research works have been done in the field of control theory and autonomic computing [29], [7]. Amongst such works, we cite the IBM MAPE-K [27] framework, that defines a reference model for autonomic control loops, where detected events are analyzed and possible actions are planned and executed using support from the internal knowledge of the system. Research works exists that also focus on developing middleware solutions to improve the quality of networking services in distributed and pervasive systems [7], [29], [28].

Despite these efforts, establishing well integrated approaches for decentralized management of highly dynamic inhomogeneous networking systems remains an open topic. In this respect, the ODDR framework is proposed to satisfy the end-node driven fault management approach [5]. The objective is to improve end-user service resilience even when the coupling between the networking devices and the available networks is weak i.e., the end-node may not have explicit knowledge of the overall network resilience and of the real-time features that the network may provide, but can still take advantage of the diversity of the networks. To achieve this objective, the ODDR needs to attentively investigate the information provided by the observed events and the information trustworthiness (e.g., computing the related measurement uncertainty [10]). More precisely, the ODDR framework merges challenges and visions from several research areas, mainly i) *dependable and real-time services* ii) *real-time autonomic network fault management*, and iii) *network hand-over approaches.*

Providing *dependable and real-time services* in future networks is a well recognized challenge, where existing focus has been on i) end-user service provisioning perspective (e.g., defining robust service provisioning techniques [17]), ii) infrastructure network perspective *(*e.g., redundant architectures and fault management [18]), and iii) ad-hoc network perspective (e.g., distributed replication [19]). The approach proposed in the ODDR is seen to complement such developments by adding awareness of end-to-end service dependability to the end-nodes themselves. This will allow the end-node to make use of the diversity and redundancy that multiple access networks and end-user service points will offer, without requiring these to offer special information as guarantees of QoS requirements.

A related field of research is *real-time autonomic network fault management* in complex networks. It has recently gained increased attention as network operators and service providers seek new approaches to manage their increasingly complex network and system architectures [18]. Identifying faults in such systems is often a significant challenge due to uncertainties (e.g., unspecified network architectures, or unreliable observations) in observing relevant events [18], [21], [6]. To aid manual and autonomous remediation, existing work is focused on improving fault diagnosis considering accuracy [23], adaptability [21], and reducing computational complexity when correlating multiple observations [22]. In addition, interesting gains of combining diagnosis-recovery have readily been demonstrated in [16] for central decision mechanisms with the objective to plan optimal recovery from faults in a server architecture. Many of these challenges are shared by the end-node approach where existing solutions may apply. In the ODDR framework, also additional gains are provided in the joined view on observation and information pre-processing, (imperfect) diagnosis and decisions of remediation (and remediation execution), additionally considering uncertainty in the observations as information useful to improve diagnosis, dynamic changes of the networking environment and planning of remediation actions based on optimization of the end-user service parameters.

The ODDR challenges and novelty bear strong similarities to the *network hand-over approaches* [20]. A central issue is the decision function to ensure minimum QoS requirements to be met while reducing unnecessary hand-overs (and possibly minimizing energy consumption). The ODDR approach can consider similar objectives, but its focus is dependability and real-time properties, introducing novelties as: i) remediation of faults on the entire end-to-end path, opposed to only on the first hop; ii) dynamically updated QoS requirements based on the end-user service state in contrast to typically fixed QoS requirements; iii) use of prior and historical knowledge to determine network dependability properties.

We are currently investigating some aspects of the ODDR, namely: i) the ODDR may evaluate the confidence in the observation and potentially use this information to improve diagnosis outcome (Section IV); ii) observation and diagnosis imperfections may be overcome by decision strategies considering end-user service state (Section V); and iii) object-oriented, automatic and online adaptation of the decision mechanisms to match the continuous evolution of the system (Section VI).

III. Design of the ODDR Architecture

A. The ODDR Components and Modules

The ODDR aims to improve the network resilience for end-user services: depending on the specific service requirements for network communication, the ODDR will operate to maintain the network communication (and consequently the end-user service that depends on the network behaviour) within the service requirements. An overall view of the ODDR architecture and of its interfaces is shown in Fig. 2 and described in details in what follows.

The ODDR framework is a middleware service composed of four different components (the four white boxes in Fig. 2): the *OPP* (*Observation & Pre-Processing*),

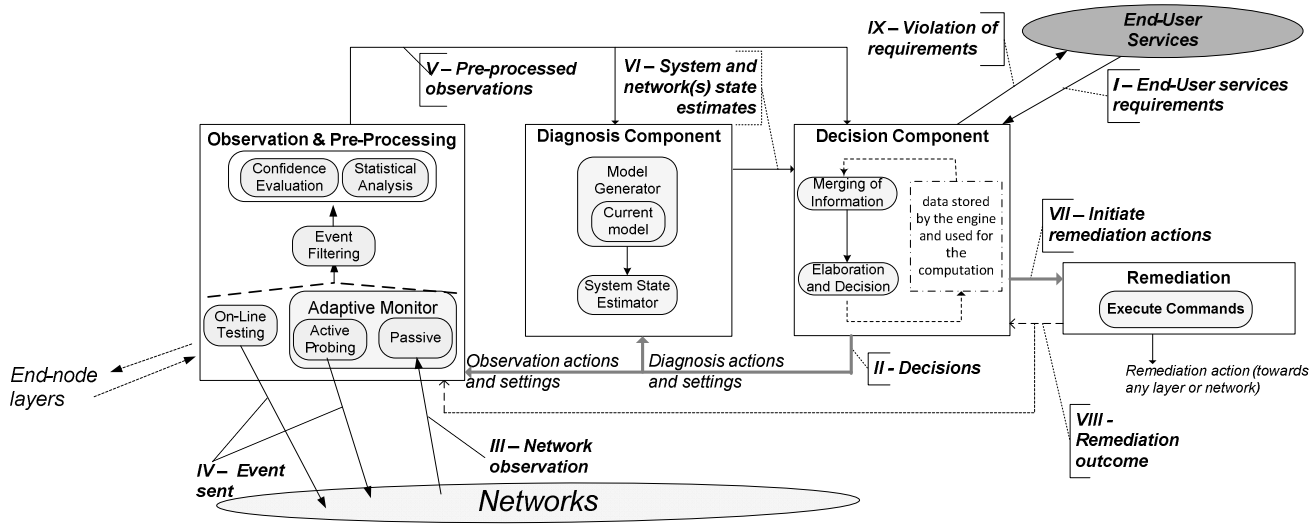

Figure 2. High-level overview of the ODDR architecture.

the *Diagnosis*, the *Decision* and the *Remediation* components.

Each component is subdivided in a set of optional modules. Different ODDR instantiations may implement a different subset of the ODDR modules, or have different module implementations and settings. Such flexibility of the framework allows to develop different instantiations depending on the available computational resources.

Optionally, depending on the hardware and system facilities, a repository or storage component may be used to store data computed and used by the ODDR components.

Observation & Pre-Processing. The Observation & Pre-Processing (OPP) component performs network observations, using i) passive and active monitoring functions (module *Adaptive Monitor*) that can be configured at run-time depending on the information required by the other ODDR components, and ii) online testing procedures (module *On-Line Testing*), that may allow to accelerate the collection of relevant data about the state of a given network and improve the effectiveness in terms of accuracy and timing for detecting or predicting anomalies. To perform this function, the OPP may interact with the various layers of the end-node.

To deal with the possible large amount of data collected, the monitoring functionalities of the OPP collect events that are successively filtered by a *Event Filtering* module, to distinguish the relevant events (e.g., because they represent deviation from correct service), from the non-relevant ones. The filtered events are then fed to two modules: the module *Statistical Analysis*, which provides statistical analysis on the data collected, and the module *Confidence Evaluation*. Regarding this last module, we note that the OPP is the measuring instrument of the ODDR, and consequently it must be developed with a particular care to collect reliable observations [6], [10]. To achieve this objective, the OPP design and implementation may take advantages of measurement theory basics (as well as well-known practice from monitoring and testing fields). For example, the monitoring and testing functionalities shall be low-intrusive, and computation of the uncertainty in the collected measurement results may help in providing confident and representative results [6].

Diagnosis component. The Diagnosis component estimates the actual system state (networks and end-node) depending on the collected observations. As many states (and particularly fault states) of the network are not directly observable (i.e., hidden), the network state estimation must rely on available observations and on a system model to provide knowledge about the hidden states. The model used could here be based on any formalism, from simple (e.g., thresholds) to complex (e.g., Bayesian Networks) ones.

Diagnosis in the scenario envisioned for the ODDR requires models that are able to online self-adapt to changes in the requirements, and that extensively use the observation collected to define the model evolution: in the ODDR, the module *Model Generator* aims to dynamically generate the new model (evolving the current model); then the module *System State Estimator* combines model and observations to compute the diagnosis outcome.

Decision component. The Decision component supervises and leads the execution of the entire ODDR framework, and takes decision on possible remediation actions. Knowing the services requirements for network communication, the Decision component is in charge of properly configuring the Diagnosis and the OPP components (e.g., setting parameters, changing intensity of online testing activities, or modifying accuracy requirements for the diagnosis component), and to identify possible remediation actions. During the service execution, the Decision component i) collects inputs from the OPP, the Diagnosis components, the end-node services, and the Remediation component, ii) using the information gathered (module *Merging of Information*), it elaborates decisions (module *Elaboration and Decision*) on possible remediation actions, or requests for different activities of the OPP and Diagnosis

(e.g., increase the frequency of the diagnosis outputs, require additional monitoring activities, or command a proper remediation action).

To prevent faults from leading to a service failure, timely remediation is vital i.e., the Decision component shall timely operate to decide the most suitable remediation action when a fault is detected, diagnosed or predicted. Consequently, efficient and real-time decision techniques are required to implement the Decision component.

Remediation component. This component performs remediation actions (module *Execute Commands*) as switching to a different wireless channel, selecting a new access network or, otherwise, modifying the communication policies. Remediation actions shall be timely executed, and their outcome needs to be monitored in order to obtain information on its outcome and characteristics for potential adaptation of the ODDR components.

B. The ODDR Interfaces

The interfaces of the ODDR are depicted in Fig. 2 and labeled using roman numbers (from I to IX).

End-node services and Decision component. The interactions of the ODDR with the end-node services are completely handled by the Decision component. The end-node services that exploit the ODDR provide their dependability requirements for the communication services (label *I - End-User Services Requirements*). The Decision component operates to meet them, sending actions and settings to the other ODDR components. The Decision component may communicate to the end-node services possible critical information, as the detection of requirements violation (label *IX - Violation of requirements*).

Decision and Diagnosis components. The Decision component communicates to the Diagnosis component the settings and actions of the diagnosis activity (label *II - Decisions*) e.g., modify/reactivate diagnosis activity to fulfill accuracy requirements. The Diagnosis component communicates its diagnosis outcome to the Decision component (label *VI - System and network(s) state estimates*).

Decision and OPP components. The Decision component communicates to the OPP the settings and actions for the observation activity e.g., increase observation activity in order to improve the confidence in the decisions, or establish new rules for the event filtering activity (label *II - Decisions*). The OPP will provide the Decision component with events detected, system state observations, and the current system parameters to be used in the decision model (label *V - Pre-processed observations*).

OPP and Diagnosis components. Pre-processed observations provided by the OPP are used as input to the Diagnosis component (label *V - Pre-processed observations*).

OPP component, the network(s) and the end-node layers. The OPP component collects events (e.g., faults or packets) by monitoring its available networks (label *III - Network observations*). The OPP contains modules for active monitoring and online testing: consequently it is able to send events (ICMP ping, application layer requests, etc) through the networks (label *IV - Event Sent*). Additionally, the OPP may interact with and use the other layers of the node.

Decision and Remediation components. The Decision component commands remediation actions, whenever needed, to the Remediation component (label *VII - Initiate remediation actions*). The Remediation component communicates to the Decision component the outcome of the remediation action (label *VIII - Remediation outcome*); the remediation outcome is provided in order to obtain information for potential adaptation of the ODDR components (including new decisions in case the remediation procedure fails).

Remediation and OPP components. The Remediation component may communicate to the OPP its outcome (label *VII - Remediation outcome*), to timely alert the OPP on possible relevant events that may otherwise result undetected (or not timely detected).

IV. Improving Diagnosis through Uncertainty in Network-Traffic Observations

One of the main challenges of the ODDR is to handle unreliable observations. In this section we present our ongoing work on attempting to improve the robustness and capabilities of diagnosis to operate under unreliable observations. How to construct a diagnosis approach that can make use of knowledge of measurement uncertainty to minimize diagnosis imperfections is explored.

We introduce the rationale of the approach and describe how it fits in the ODDR framework. Next, we introduce an example scenario and finally present intermediate results based on a detailed system level simulation analysis. Further details can be found in [13].

A. Measurement Uncertainty in Diagnosis

The observations collected from highly dynamic and complex networks are typically unreliable (e.g., noisy [22]). The measured values that are affected by non-negligible measurement errors may lead to significant imperfections of the diagnosis estimates and thereby affect the end-user service resilience [6], [10]. To evaluate and mitigate possible measurement errors, the body of knowledge of metrology (measurement theory [10]) proposes rules and good practices: in particular, measurement uncertainty (or simply *uncertainty*) provides quantitative information on the dispersion of the quantity values that could be reasonably attributed to the measurand [10]. Uncertainty has to be evaluated according to conventional procedures, and is usually expressed in terms of a *confidence interval*, that is a range of values where the measurand value is likely to fall.

As examples we may consider the measurement of network delay (one-way or round-trip time, e.g. used for congestion control in TCP). Such delays may be affected by time measurement uncertainties such as drifting clocks and clock resolution [15], clock synchronization uncertainty [15], destination host processing and queuing time (e.g. in

case of application based ping) [11], intrusiveness of the measuring system [10] or delays on the sender node [11].

Overall, diagnosis approaches are sought to improve the performance of diagnosis given uncertain and unreliable observations as in [23] using Hidden Markov Models (HMMs), in [25] considering adaptive probing, and in [22] based on correlation of multiple observations. Combining the disciplines of metrology and diagnosis may allow to improve the robustness of system diagnosis to unreliability caused by the observation process. Similar approaches based on HMMs have been applied in the field of speech recognition to improve robustness in noisy environments [26].

B. Scenario, Faultload and Workload

A general example of inclusion of the ODDR in an end-node is depicted in Fig. 3. A mobile node uses services in an infrastructure network where in the end-to-end path various faults may occur. The access network diversity offers the possibility of fault remediation by access network selection. Diagnosing the right cause (e.g. contention or congestion) will have a significant impact on the decision outcome.

An end-user service considered in the example is a reliable transfer of a certain amount of data within a critical time deadline requirement. This generic end-user service case covers various end-user services such as: i) upload of patient examination results to the emergency room during an emergency response; ii) streaming of video/audio (with progressive download) where an amount of data must be transferred in time to avoid buffer underrun; and iii) file download/upload within time constraints to ensure a certain productivity in industrial applications. The influence of network faults may here lead to service failure.

A detailed simulation model of the example scenario has been developed. For simplicity, only a single congestion fault assumption is considered, which is sufficient to study and illustrate: i) how observation uncertainty and mitigation options impact diagnosis, and ii) how imperfect diagnosis should be considered in the decision process (section V). A network state can be defined from a given packet loss rate and network delay. Measurement based studies in [30] have shown that infrastructure network paths may in reality be characterized by relatively few states (1-4). In correspondence, a two-state congestion fault model is considered under the following assumptions. A link in the infrastructure experiences varying cross-traffic. In a *normal state* the cross-traffic amount is low leading to a negligible amount of congestion. In a *fault state* the cross-traffic is at a level where packet losses are high causing a significant reduction in the end-to-end connection throughput. It is assumed that the infrastructure networks are independently alternating between these two states over time according to a geometric ON-OFF model.

This type of fault can be diagnosed starting from round-trip time (RTT) observations. We consider that the RTT between the end-node and the service provider may be altered due to network disturbances (noise) as queuing delays in other queues than the congested one [11]. As a basic approach we represent this noise synthetically in our simulation environment adding Gaussian noise to the round-trip delays collected.

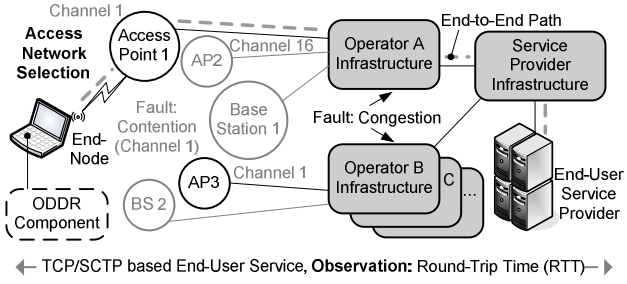

Figure 3. Example network scenario.

C. Observation Collection and Diagnosis in the ODDR

In Fig. 4 a general view is presented on the ODDR framework developed to compute measurement uncertainty and use it to improve diagnosis. The focus is on the OPP and the Diagnosis components. Thus, the Decision and Remediation components are here considered as simple stub implementations where a remediation action (a fail-over) is initiated when a network fault state is estimated.

The Passive Monitor within the OPP collects RTT observations from existing traffic of an SCTP stream. In the Statistical Analysis module a basic filtering is performed to obtain mean estimates from fixed-size time windows (which may consist of a varying amount of RTT samples due to SCTP congestion control).

A central role is devoted to the Confidence Evaluation. Using the most recent window of data received, the module Confidence Evaluation provides to the Diagnosis component the triple $[X, X^{LOW}, X^{UP}]$, where X is the point estimate and X^{LOW} and X^{UP} are the lower and upper uncertainty bounds. Correspondingly to the information received, the Diagnosis component produces a state estimate.

The diagnosis mechanism implemented in the Diagnosis component considered in this work is based on the HMM formalism [12]. In brief a HMM consist of the following elements: i) a transition probability matrix A for the hidden

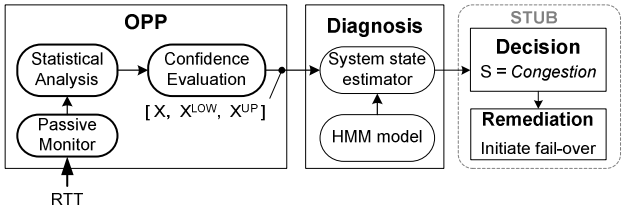

Figure 4. The ODDR considering combined measurement uncertainty and diagnostic activity.

state Discrete Time Markov Chain (in our case representing the network), ii) an observation probability matrix B describing the observation distributions over states in A, and iii) π the initial state distribution. We present two different models, (H) and (Hu). (H) corresponds to a model without considering uncertainty. The model is based on a fixed discretization of RTT values to give a discrete distribution in B. (Hu) is the same as (H) but uses the information on measurement uncertainty to update B for each observation provided by the OPP. In practice, this is commenced by a dynamic discretization approach: details may be found in [13]. Applying a HMM for diagnosis, it can be used to derive the probability δ_{FAULT} of being in the fault state given an observation sequence. A threshold on such probability δ_{FAULT} defines when S = *congestion* is considered.

D. Simulation Results

To highlight the potential of this approach, example results are presented. System level simulations of the scenario in Fig. 3 have been conducted in the network simulation tool ns-2 [24] to obtain real RTT traces (2000 simulation runs of 300 seconds each) for a given fault scenario (defining severity and mean normal/fault cycle times). Subsequently, these traces have been parsed through an implementation of the ODDR (in MATLAB) to provide sequences of state estimates. From these sequences, mean diagnosis state estimation metrics are derived. In our simulative example, the uncertainty bounds X^{UP} and X^{LOW} are computed assuming that the Confidence Evaluation has knowledge on the noise characteristics (Gaussian noise, parameters $N(0,10^2)$).

Assuming the basic decision function used in the ODDR considered in this section, in Fig. 5 these metrics are the Probability of Remediation on a True Alarm P_{RTA} and the Reaction Time. Reaction Time is the time from a fault has occurred until it is potentially diagnosed (in a single fault cycle). Example results are depicted for (H) and (Hu), considering both non-noisy RTTs (clean RTTs), and RTTs affected by Gaussian noise (noisy RTTs). For the clean RTTs, the (H) and the (Hu) model behaves the same; consequently only one trace is shown in Fig. 5, with label *(H) Clean obs*. For the noisy RTTs, the results using the (H) model are labeled as *(H) Noisy Obs* and the results using the (Hu) model are labeled as *(Hu) Noisy Obs*.

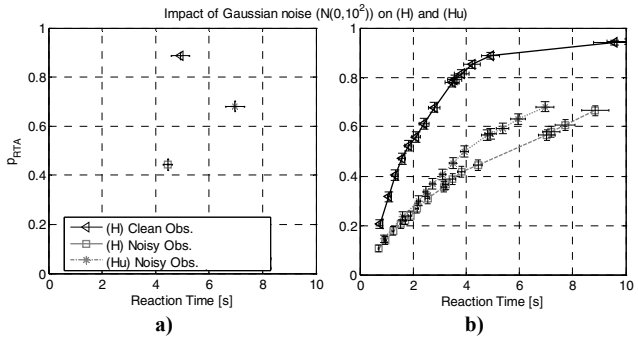

Figure 5. Results on using information on measurement uncertainty to improve diagnosis, considering a) a fixed threshold of 0.99 on δ_{FAULT} and b) different thresholds.

In Fig. 5a a fixed setting of the threshold on δ_{FAULT} is considered (as diagnosis might be implemented in reality). As expected, using clean RTTs the best results are achieved. Considering noisy RTTs, we note that in the (H) model P_{RTA} drops significantly, without providing significant changes in the Reaction Time. In case of the (Hu) model using noisy RTTs, the drop of P_{RTA} is significantly smaller than using the (H) model, at the cost of a slightly increased Reaction Time.

Fig. 5b depicts instead the obtainable trade-offs between the two diagnosis metrics for different thresholds on δ_{FAULT}. Again, using clean RTTs the best results are achieved. Considering noisy RTTs, P_{RTA} drops to around the half in the (H) model for a given Reaction Time. Instead, (Hu) is shown to enable better trade-off options, yet, still far from the clean observation case.

From previous studies of the reliable transfer service case [5] it has been seen that reliability is most sensitive to a high amount of false alarms compared to changes in Reactivity Time. In this sense the diagnosis with measurement uncertainty information can provide improved robustness to uncertainties. However, additional work is relevant to study the impact on other end-user services e.g., based on real-time traffic.

V. Decisions Under Imperfect Diagnosis

A central challenge of end-node driven fault management is to create useful reliability models enabling derivation of good decision strategies under uncertain observations and imperfect diagnosis. In this section we describe a model for the Decision component that allows selecting remediation policies that will improve service reliability despite imperfect and un-trusted Diagnosis and OPP components as depicted in Fig. 6. The discussion presented is based on an earlier work [5] that investigates good remediation policies (the rules to select the remediation actions) for an end-node service under unreliable diagnosis.

A. Decisions in the ODDR and The File Upload Scenario

From a general view the Decision component consists of a (prediction) model used to infer best remediation policies for a given set of system parameters. This model may be constructed offline (prior to service execution) or online as system parameters change. In this section, we consider models that are constructed offline whereas Section VI discusses online construction.

In studying a specific decision problem, we consider the same end-node service, scenario, faultload and workload introduced in Section IV. The ODDR instantiated for this case study aims to optimize the probability Ω of completing a file transfer upload in time, using good remediation

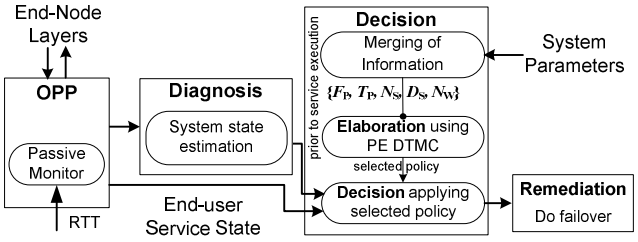

Figure 6. Improving Decisions policies to supply for simple and not trusted OPP and Diagnosis components.

decisions to tolerate intermittent faults, wrong diagnosis estimates, and late occurring faults. The ODDR instantiation developed to address the challenges considered in this section is as follows (Fig. 6).

In the OPP, a simple passive monitoring mechanism measures i) RTTs from existing traffic, and ii) current service status (e.g., 50% of the file has been uploaded). The Diagnosis component performs periodic estimations on network state. It is characterized by imperfections and contains options to provide different trade-off of such imperfections (e.g., as exemplified in Fig. 5b). The Remediation component simply performs access network fail-overs whenever requested by the Decision component.

The Decision component is structured as follows. The module Merging of Information takes as input system parameters as: i) information on Diagnosis component characteristics, ii) end-user service requirements, and iii) information on available remediation networks and their a-priori state probabilities. The system parameters are collected in the module Merging of Information to create a Policy Evaluation model based on Discrete Time Markov Chain (PE DTMC). This model can be elaborated (in the Elaboration and Decision module) to compute good or optimal decision policies for the selected service. In particular, the Elaboration part enables to compare potential remediation policies such as i) do nothing, ii) at diagnosed congestion state, perform fail-over, or iii) at diagnosed congestion state and when a minimum period of time ($\gamma_{MinTime}$) has passed since the initiation of the file transfer, perform fail-over. The latter is an example where end-user service state information is used in the decision.

The Decision part of the module Elaboration and Decision applies the selected decision policy.

B. Results and Further Remarks

Studying such types of policies has helped to provide relevant insights into the impact of different imperfect diagnosis characteristics under different policies on an end-user service. Our findings in [5] are that: i) the properties of the diagnosis components have a significant impact on the probability Ω; ii) for certain trade-off settings of the Diagnosis component (that generates a high amount of false alarms), decision policies where fail-over is performed at fault diagnosis can perform significantly worse than the no fail-over policy; iii) finally, using state information in the policy combining state estimation with $\gamma_{MinTime}$ can in all the studied cases help improve reliability to perform significantly better than the no fail-over policy. Even for this basic two-network setup, it is shown how reliability gains may be achieved when the Decision component uses knowledge of the imperfect diagnosis model and of the end-user service state.

Model based decisions, clearly, introduce significant complexity to the ODDR framework. A fundamental challenge of the studied approach is to ensure general and lightweight models which are basic to parameterize and solve. The PE DTMC model studied for this scenario has, thus, been designed using simple birth-chains and minimal representations of diagnosis states. Despite these approaches (simplifying details of the end-user service operation and fault impacts), the proposed models have in comparison to detailed simulation analysis been consistent with respect to policy outcomes. In ongoing work the studied model is extended to multi-fault, multi-remediation option scenarios and to include active probing approaches for improved diagnosis and remediation options. In such complex settings (under imperfect diagnosis) a model based approach is expectedly superior to simpler heuristics when considering the problem of optimizing dependability and minimizing overhead from unnecessary remediation and probing actions. In related works, benefits of model based decisions are readily demonstrated for hand-over techniques [20] and efficient fault recovery planning in infrastructure networks [16].

VI. Adaptive Decisions through Object-Oriented Online Model Generation and Composition

The Decision component should take decisions based on a correct and up-to-date system model. The Decision component designed in Section V is using decision policies that are computed prior to service execution; however the ODDR is expected to operate in a ubiquitous and highly changing system environment, leading to the additional challenge of finding good/optimal decisions in a (continuously) evolving system. In this section we present our envisioned approach to this problem and consequent ongoing work.

A. Decision Processes in Resilient Networking Systems

In general, building computer systems which are capable of taking autonomous decision on their configuration is a well recognized issue. In the past, several research programs have been devoted to it and to its formalization [7].

In the field of networking systems, middleware solutions have been proposed (e.g., [9]) to optimize real-time and dependability requirements of network applications, through the use of distributed objects and resources. The novelty introduced by the envisioned ODDR decision process lies in the possibility of a decentralized approach where optimization and decision is left to the end-node, which may

not be aware of the resilient and real-time features that the network may provide (if any).

B. Object-Oriented Online Model Generation

In the ubiquitous environment in which the ODDR is supposed to operate, it is not feasible to build a model that takes into account all the possible evolutions of the system. Consequently, an online model generation approach [16] is here envisioned and proposed, where an adaptation controller (managed by the Decision component) maintains an internal model of the system being controlled, and periodically updates and solves it based on the collected measures.

The system model should be able to accomplish the two following tasks: i) *track* the system evolution, taking into account the changes that may occur in its configuration (this involves mapping the current system configuration to an appropriate model state, and moving to another state when the system configuration changes); and ii) *predict* the system evolution, using the model of the system and the current system configuration. Based on this prediction the policy that maximizes the service resilience is derived.

The system model should contain only a reduced set of states from all those possible for the system; these states should be limited to the current state and a set of *near* states (where *near* needs to be defined using a *distance function* on system configurations). For each of this states the future system evolution is evaluated through the prediction model(s), in order to estimate the application reliability corresponding to the different possible choices. When the system moves outside the state-space modeled by the tracking model, a new model generation step is triggered, which leads to the creation of a new tracking model and the subsequent evaluation of the prediction models.

We propose a compositional modeling approach to automatically generate models for different system configurations, starting from a small number of basic (parametric) atomic models. This reduces the number of models to be generated, that otherwise would risk to be exaggeratedly wide. The envisioned approach is based on the Stochastic Activity Networks (SAN [14]) high-level modeling formalism as a basis to exploit an Object-Oriented (OO) approach for model construction. In particular, one of the features of SAN is the ability to create the system's model by composition of atomic submodels, through the *join* and *replicate* operators.

The compositional modeling is exploited by the use of parametric models, following an approach which resembles OO programming and which is driven by the same needs: abstraction, modularity, encapsulation, reuse of components. After identifying the basic "building blocks" of the modeling domain which is of interest, the respective parametric atomic models (the classes, in OO terminology) are created. Using multiple instances of these atomic models with different parameters, and following specific compositional rules, a model for every possible system configuration can be easily (and automatically) obtained, like in OO programming classes may be instantiated with every allowed values of their attributes. Actually, we are performing a similar operation to object instantiation in OO terminology.

This approach leads also to the same advantages of object oriented programming: less "code" to be written, easier model maintainability, and higher level reasoning in the model construction phase.

The ODDR instantiation proposed to address these argumentations is depicted in Fig. 7. The OPP and Diagnosis components provide the model parameters for the current system configuration. Merging the input parameters, the atomic models and the composition rules (module Merging of Information), both the tracking and prediction models for the current configuration are generated (module Elaboration and Decision). Prediction models are used to perform policy evaluation i.e., select the optimal actions for the configuration (states) which have been taken into account in the tracking model. The tracking model validates the current parameters based on the input data received from the OPP component and, if necessary, updates the current state of the model to match the current system configuration, and commands the Remediation component to initiate the actions given by the pre-evaluated policy. When the system changes in a way such that the new state falls outside of the states modeled in the tracking model, a new online generation step is triggered and both the tracking and the prediction models are generated (or simply updated) from the template atomic models and composition rules. The adaptation policy specifies to the Remediation component how the system should adapt to the changes.

To concretize our approach, our research is currently focusing on two main directions: i) properly identify the basic logical blocks of the overall model, in order to derive the corresponding template atomic models, and ii) defining the distance function.

VII. CONCLUSIONS

In this paper we introduced the framework *ODDR* (*Observation, Diagnosis, Decision, Remediation*) for self-adaptive and online observation, diagnosis and consequent decision on possible remediation actions in highly-adaptive and heterogeneous networks. The ODDR is a middleware

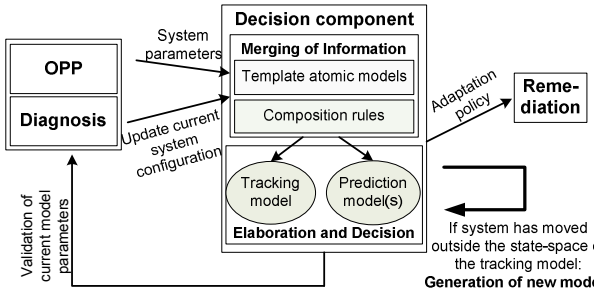

Figure 7. Object-Oriented online model generation and adaptation of the Decision component.

service that aims to improve the resilience of network communication for end-user services exploiting an approach based on end-node driven fault management.

In this paper we presented the overall ODDR architecture and three directions that we are currently investigating towards the ODDR development. The first one investigates the possibility to improve the network diagnosis using measurement uncertainty of network observations. The second is devoted to the analysis of online decision and remediation strategies that supply to missing observations or unreliable diagnosis. Finally, the last one proposes an approach to develop automatic and online adaptation of the decision mechanisms to match the continuous evolution of the system and the outcome of possible remediation action.

We plan to further explore and refine the three different topics described and to proceed to integrate the different techniques (those already developed and here discussed and those that we are going to develop starting from the observations in this work) in a unique ODDR instantiation.

ACKNOWLEDGMENT

This work has been partially supported by the European Project FP7-IST-234088 ALARP, by the PRIN Project Assuring and Assessing Resilience of Large Scale Critical Infrastructures funded by the Italian Ministry of Research, by Tieto IP solutions, Viby J, Denmark, and by the Telecommunications Research Center Vienna (FTW), Wien, Austria.

REFERENCES

[1] B. Hughes, R. Meier, R.Cunningham, and V. Cahill, “Towards real-time middleware for vehicular ad hoc networks”, VANET ’04, ACM, 2004, pp. 95-96.

[2] HIDENETS - HIghly DEpendable ip-basedNETworks and Services - FP6-IST-2004-26979, http://www.hidenets.aau.dk/.

[3] T. Madsen (editor) et al., “D3.1 - wireless communication architecture and protocols,” Tech. report, 2008, FP6 European project SAFEDMI Deliverable 3.1.

[4] M. Satyanarayanan, "Pervasive computing: vision and challenges," Personal Communications, IEEE , vol.8, no.4, pp.10-17, Aug 2001.

[5] J. Grønbæk, H.-P. Schwefel, and T. Toftegaard, “Model based evaluation of policies for end-node driven fault recovery,” Proc. DRCN 09, 2009.

[6] A. Bondavalli, A. Ceccarelli, L. Falai, and M. Vadursi, “Foundations of measurement theory applied to the *evaluation of dependability attributes,”* Proc. IEEE/IFIP DSN, IEEE Computer Society, Washington, DC, 2007, pp. 522-533.

[7] M.C. Huebscher, and J. A. McCann, “A survey of autonomic computing - degrees, models, and applications,” ACM Comput. Surv., vol. 40, no. 3, Aug. 2008. pp. 1-28.

[8] J. Gade, M. Dahl, P. Thorhaard and F. Knudsen, “AMPHItm Ambulance Record Keeping System,” Journal of Clinical Monitoring and Computing, vol. 20, 2006, pp. 117-144.

[9] D.C. Schmidt, “Middleware techniques and optimizations for real-time, embedded systems,” Proc. of the 12th Int. Symp. on System Synthesis, 1999. IEEE Computer Society, Washington, DC, 12.

[10] BIPM, IEC, IFCC, ISO, IUPAC, and OIML, “ISO international vocabulary of basic and general terms in metrology (VIM),” 3rd ed., 2008.

[11] J. Aikat, J. Kaur, F. D. Smith, and K. Jeffay, “Variability in TCP round-trip times,” Proc. ACM SIGCOMM Conference on Internet Measurement, New York, NY, USA: ACM, 2003, pp. 279-284.

[12] L. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” Readings in speech recognition, vol. 53, no. 3, 1990, pp. 267-296.

[13] J. Gronbaek, H.P. Schwefel, A. Ceccarelli, and A. Bondavalli, “Improving robustness of network fault diagnosis to uncertainty in traffic based observations”, Technical Report, 2010.

[14] W.H. Sanders, J.F. Meyer, “Stochastic activity networks: Formal definitions and concepts,” Lectures on Formal Methods and Performance Analysis, vol. 2090 of LNCS, Springer, 2001, pp. 315-343.

[15] A. Bondavalli, A. Ceccarelli, and L. Falai, “Assuring resilient time synchronization,” Proc. IEEE SRDS, Washington, DC, USA: IEEE Computer Society, 2008, pp. 3-12.

[16] K. R. Joshi, W. H. Sanders, M. A. Hiltunen, and R. D. Schlichting, “Automatic Model-Driven Recovery in Distributed Systems,” Proc. of the 24th IEEE Symposium on Reliable Distributed Systems, IEEE Computer Society, Washington, DC, 2005, pp. 25-38.

[17] T. Renier, H.P. Schwefel, M. Bozinovski, K. Larsen, R. Prasad, and R. Seidl, “Distributed redundancy or cluster solution? An experimental evaluation of two approaches for dependable mobile Internet services,” Lecture Notes in Computer Science, vol. 3335, Springer, 2005, pp. 33-47.

[18] C.R. Kalmanek et al., “DarkStar: using exploratory data mining to raise the bar on network reliability and performance,” Proc. DRCN, 2009.

[19] E.V. Matthiesen, O. Hamouda, M. Kaaniche, and H.P. Schwefel, “Dependability evaluation of a replication service for mobile applications in dynamic ad-hoc networks,” Lecture Notes in Computer Science, vol. 5017, Springer, 2008, pp. 171-186.

[20] E. Stevens-Navarro, Y. Lin, and V. Wong, “An MDP-based vertical handoff decision algorithm for heterogeneous wireless networks,” IEEE Transactions on Vehicular Technology, vol. 57, no. 2, 2008, pp. 1243- 1254.

[21] P. Bahl, R. Chandra, A. Greenberg, D. Maltz, and M. Zhang, “Towards highly reliable enterprise network services via inference of multi-level dependencies,” Proc. SIGCOMM’ 07, ACM Press New York, NY, USA, 2007, pp. 13-24.

[22] M. Steinder and A. Sethi, “Probabilistic fault localization in communication systems using belief networks,” IEEE/ACM Transactions on Networking, vol. 12, no. 5, 2004, pp. 809-822.

[23] A. Daidone, F. Di Giandomenico, S. Chiaradonna, and A. Bondavalli, “Hidden markov models as a support for diagnosis: formalization of the problem and synthesis of the solution,” Proc. IEEE SRDS, 2006, pp. 245-256.

[24] NS-2 - The Network Simulator, http://www.isi.edu/nsnam/ns/, 2005.

[25] M. Natu and A. Sethi, “Probabilistic Fault Diagnosis Using Adaptive Probing,” Proc. of DSOM 2007, San José, CA, USA, October 29-31, 2007. Springer, 2007.

[26] Liao, and M.J.F. Gales, “Issues with uncertainty decoding for noise robust automatic speech recognition,” Speech Communication, Elsevier, 2007, pp. 265-277.

[27] IBM, “An architectural blueprint for autonomic computing,” Technical Report, Third Edition, 2003.

[28] R.E. Schantz (editor) et al., “Quorum distributed object integration (QuOIN),” Technical Report, 2002.

[29] S. Dobson et al., “A survey of autonomic communications,” ACM Trans. Auton. Adapt. Syst., vol. 1, no. 2, 2006, pp. 223-259.

[30] K. Salamatian and S. Vaton, “Hidden markov modeling for network communication channels,” SIGMETRICS Perform. Eval. Rev., vol. 29, no.1, 2001, pp. 92-101.