Proc. 23rd Army Science Conference, Orlando, FL (Dec. 2002). **Received Best Paper Award.**

FAULT LOCALIZATION AND SELF-HEALING MECHANISMS FOR FCS NETWORKS[1]

L. Kant*, A.S. Sethi+ and M. Steinder+
*Telcordia Technologies, 445 South Street, Morristown, NJ 07960, USA
+University of Delaware, Dept. of Computer and Information Sciences, Newark, DE 19716, USA
Contact Authors: L. Kant; lkant@research.telcordia.com; Tel: (973) 829-2879; Fax: (973) 829-5888
A.S.Sethi; sethi@cis.udel.edu; Tel: (302) 831-1945; Fax: (302) 831-8458

ABSTRACT

In the dynamic environment of mobile ad-hoc battlefield (FCS) networks, it is important to have an efficient fault management (FM) system capable of performing dynamic and rapid fault localization and providing appropriate self-healing (service survivability) to mission-critical applications in a timely and efficient manner. This paper presents a multi-layer fault localization mechanism using Bayesian techniques for narrowing down the fault/problem on hand. It also proposes appropriate cost-performance-complexity sensitive self-healing/service survivability mechanisms for FCS networks.

1. INTRODUCTION

FCS networks (i.e., the next generation of tactical and strategic battlefield networks) are envisioned to offer a highly automated, secure, survivable and novel paradigm of battlefield operations [Sass02, USA02]. An extremely critical aspect of FCS networks is the presence of an efficient fault management (FM) system that is capable of performing dynamic and rapid fault localization and providing appropriate self-healing (service survivability), particularly to mission critical applications in a timely *and* efficient manner. While fault localization/root cause analysis and self-healing for mobile ad-hoc networks are, in general, very challenging problems by themselves, the problems are compounded even further in FCS networks due to the following: (a) Ad-hoc nature coupled with mobile infrastructure of underlying network. (b) Random/sporadic failures due to hostile and/or unintentional attacks. (c) Presence of multiple and possibly correlated failures. (d) Presence of random soft failures (due to the stochastic nature of underlying network) in addition to hard failures (due to the more deterministic cases of equipment malfunction). (e) Critical need to distinguish between transient and non-transient behavior. (f) Cope with non-determinism. (g) Obtain solutions that work in real-time - not NP hard.

This paper is aimed at providing the critically needed fault localization and self-healing/survivability for FCS networks. In particular the contributions of this paper are as follows. First, we present fault localization mechanisms that can narrow down the fault/problem on hand. More specifically, we describe a multi-layer model that uses Bayesian techniques to capture the dependencies that may exist between entities in multiple network nodes and in multiple protocol layers at those nodes; we also outline algorithms to perform the desired FCS-network fault localization and briefly present some simulation results. Next, we propose appropriate cost-performance-complexity sensitive self-healing/service survivability mechanisms for FCS networks. In light of the fact that wireless resources are very expensive, the proposed mechanism is sensitive to the cost aspect. However, due to the presence of a variety of applications with varying survivability requirements (e.g., mission critical applications require very quick and assured delivery while non-mission critical and non-real time applications are tolerant to some delays and losses), the proposed mechanisms are sensitive to the performance/survivability requirements of the various applications. Additionally, the proposed self-healing mechanisms are adaptive/dynamic and policy-based, and can be implemented within the rapidly emerging policy-based network management framework. Finally, since a complex self-healing/fault tolerance mechanism will defeat its own purpose, especially in a battlefield environment where deployment-ease is an extremely important concern, the proposed mechanisms are sensitive to the complexity aspects and lend themselves to easy and automated (a highly desired feature in the FCS-network environment) implementation.

2. FAULT LOCALIZATION FOR FCS NETWORKS

FCS networks, being wireless mobile and ad-hoc, possess many unique characteristics because of which most existing fault diagnosis techniques cannot be directly used. Existing traditional fault diagnosis methodology mostly concentrates on detecting, isolating,

[1] Prepared through collaborative participation in the Collaborative Technology Alliance (CTA) Communications and Networks (C&N) sponsored by the U.S. Army Research Laboratory under the Federated Laboratory Program, Cooperative Agreement DAAD19-01-2-0011.

and correcting faults related to network connectivity [Jakobson93, Katzela95]. Such diagnosis is focused on lower layers (physical and data-link layers) [Nygate95, Yemini96], and its major goal is to isolate faults related to the availability of network resources, such as broken cable, inactive interface, etc. It is common to use event correlation techniques to focus on the cause of a problem when a manager may receive thousands of alarms from different sources resulting from a failure. Since availability problems are relatively rare, most event correlation techniques existing today assume that only one fault may exist in the system at any time and do not attempt to detect multiple simultaneous faults. In addition, these techniques frequently use a deterministic model, which assumes that all dependencies and causal relationships are known with 100% certainty.

In the dynamic environment of mobile ad-hoc battlefield (FCS) networks, diagnosis may no longer be constrained to the lowest layers of the protocol stack. On the contrary, fault diagnosis has to reach through the transport and application layers into the service layer. Since upper layers are necessarily dependent on lower layers in order to provide their services, the fault management system should integrate fault diagnosis across multiple layers. As part of the ARL-sponsored CTA Consortium on Communications and Networks, we have developed a research methodology for fault diagnosis in FCS networks. Our techniques are designed to focus on failures that affect network services and their use by applications in addition to the more commonly-studied hardware failures. We have developed a multi-layer model that uses Bayesian techniques [Heckerman95, Pearl88] to capture the dependencies that may exist between entities in multiple network nodes and in multiple protocol layers at those nodes. In the following subsections, we describe this hierarchical model and some Bayesian algorithms that operate on this model to perform fault correlation. We also outline a novel incremental algorithm for hypothesis updating and then briefly present some simulation results on the performance of all the algorithms.

2.1 Layered Model for Alarm Correlation

For the purpose of fault diagnosis, communication systems are frequently modeled in a layered fashion imitating the layered architecture of the modeled system [Gopal00, Yemini96]. This approach provides a natural abstraction of the modeled system's entities, reusability of the model's modules, and ability to divide the fault management task into separate, simpler subtasks. Because of fault propagation, the effects of an abnormal operation of functions or services provided by lower layers may be observed in higher layers. Fault management systems model fault propagation by representing either causal relationships among events or dependencies among system entities [Gopal00].

In the layered fault model, the definition of entity dependencies is based on real-life relationships between layers on a single host and among network nodes communicating within a single protocol layer. The layered model proposed by us for FCS networks, based on the model proposed in [Gopal00], divides the fault model components into *services* and *functions*. A service offered by protocol layer L between nodes a and c ($Service_L(a,c)$) is implemented in terms of layer L functions on hosts a and c ($Network\ Functions_L(a)$ and $Network\ Functions_L(c)$), and services that layer L-1 offers between hosts a and c. Layer L functions on node a depend on layer L-1 functions on node a. The recursive dependencies among services and functions constitute a dependency graph as presented in Figure 1.

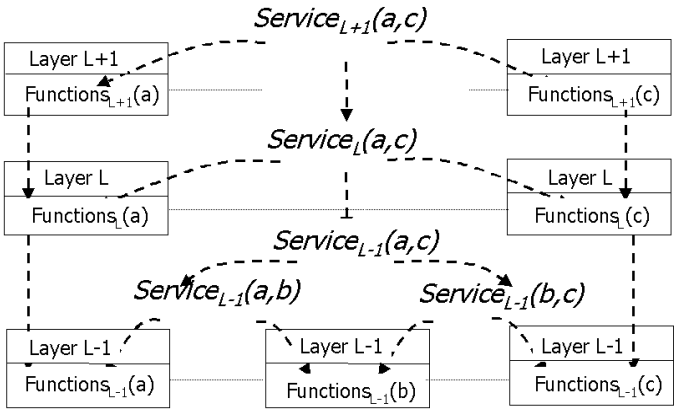

Figure 1: Layered Model

The general dependency graph template obtained from services, protocols and functions in different layers provides a macro-view of the relationships that exist in the system. To incorporate the micro-view of the relationships within particular model components, the layered model should be further refined to include possibly complex relationships within services and functions in the same layer. In particular, an end-to-end service offered by layer L between hosts a and c is implemented in terms of multiple host-to-host services offered by layer L between subsequent hops on the path of a layer L packet from node a to node c (such as $Service_{L-1}(a,c)$ in Figure 1). Ability to reason about failures observed in an end-to-end service, i.e., symptoms, and trace them down to particular host-to-host service failures, i.e., faults, is a feature of our model that is important to its applicability to fault diagnosis in FCS networks.

With every dependency graph node, we associate multiple failure modes that represent availability and performance problems pertaining to the service or function represented by the dependency graph node. We have included the following failure modes that are likely to exist in real-life FCS systems: F_1: service/function ceases to exist (e.g., a cable connection

is broken), F_2: service/function introduces unacceptable delay (e.g., one of the host-to-host links is congested), F_3: service/function produces erroneous output (e.g., bit errors are introduced in a link between routers), and F_4: service/function occasionally does not produce output (e.g., packets are lost due to buffer overflow). The micro-model of $Service_L(a,c)$ or $Network\ Function_L(a)$ defines which failure mode occurs in $Service_L(a,c)$ or $Network\ Function_L(a)$ when a particular failure mode occurs in a service or function on which $Service_L(a,c)$ or $Network\ Function_L(a)$ depends. To create a micro-model for $Network\ Function_L(a)$, the knowledge of its definition and implementation is required. The micro-model of $Service_L(a,c)$ is built based on the knowledge of the network protocol used to provide $Service_L(a,c)$. For example, knowing that layer L protocol implements an error detection mechanism, one can predict that erroneous output produced by $Service_{L-1}(a,b)$ (condition F_3) results in data loss in $Service_L(a,b)$ (condition F_4). When layer L does not implement an error detection mechanism, condition F_3 in $Service_{L-1}(a,b)$ results in condition F_3 in $Service_L(a,b)$.

Uncertainty about dependencies among communication system entities is represented by assigning probabilities to the links in the dependency or causality graph [Katzela95, Kliger95]. Some commonly accepted assumptions in this context are that (1) given fault *a*, the occurrences of faults *b* and *c* that may be caused by *a* are independent, (2) given occurrence of faults *a* and *b* that may cause event *c*, whether *a* actually causes *c* is independent of whether *b* causes *c* (the OR relationship among alternative causes of the same event), and (3) root faults are independent of one another. This dependency graph can then be transformed into a belief network, which is a directed acyclic graph with certain special properties [Heckerman95]. Given an evidence set (partial assignment of values to variables represented in a belief network), belief networks are used to make four basic queries: (1) belief assessment, (2) most probable explanation, (3) maximum a posteriori hypothesis, and (4) maximum expected utility. The first two queries are of particular interest in the context of fault correlation. The *belief assessment* task is to compute the probability that some variables possess certain values given the evidence set. The *most probable explanation (MPE)* task is to find a complete assignment of values to variables in a way that best explains the observed evidence. It is known that these tasks are NP-hard in general belief networks. To the best of our knowledge, no approximation has been proposed that works well for all types of networks. We focus on a class of belief networks representing a simplified model of conditional probabilities called *noisy-OR gates* [Pearl88]. The simplified model contains binary-valued random variables. The noisy-OR model associates an inhibitory factor with every cause of a single effect and assumes that they are all independent. The effect is absent only if all inhibitors corresponding to the present causes are activated. Belief assessment in polytrees with the noisy-OR model has polynomial complexity, which makes it attractive to use with our problem as an approximation schema.

2.2 Algorithms and Simulation Results

We now briefly outline three Bayesian algorithms to find the best symptom explanation with causal dependencies between events represented by dependency graphs transformed to belief networks as described in Section 2.1. These algorithms are called *bucket-tree elimination*, *iterative belief propagation in polytrees*, and *iterative MPE in polytrees*. Iterative belief propagation is derived from the original algorithm by [Pearl88] and is augmented by us to provide a complete symptom explanation hypothesis rather than the marginal posterior probability distribution provided by the original algorithm. For iterative MPE, an approximation is proposed that allows polynomial-time complexity to be achieved. We also describe an *incremental hypothesis updating* algorithm that has lower computational complexity than the Bayesian algorithms. Details of all these algorithms may be found in [Steinder01, Steinder02a, Steinder02b].

Bucket elimination is one of the most popular algorithmic frameworks for computing queries using belief networks. For the purpose of fault localization we use MPE query, for which this algorithm is exact and always outputs a solution. We consider it the optimal algorithm for computing the explanation of the observed symptoms. The computational complexity of the algorithm in bipartite graphs representing the problem of end-to-end service failure diagnosis is bound by $O(n^2e^n)$.

Iterative belief propagation in polytrees utilizes a message-passing schema in which the belief network nodes exchange messages called *lambda* and *pi* messages. These messages are sent by a node to its parent or child depending on computations of marginal posterior probabilities of the entire body of evidence in specified sub-graphs. Based on the messages received from its parents and children, each node then computes its belief that the variables have certain values given the entire evidence. The algorithm starts from the evidence node and propagates the changed belief along the graph edges through computations in every visited node. In noisy-OR gate belief networks, these computations may be evaluated in linear time with respect to the number of neighbors that a node has. We have adapted this iterative belief propagation algorithm to the problem of fault localization with fault models represented by bipartite graphs. In this application, we perform one traversal of the entire graph for every observed symptom. For every symptom we define a different ordering that is equivalent to the breadth-first order started in the node representing

the observed symptom. In order to use this algorithm for fault correlation, we must run the inferences again for those nodes that correspond to faults with beliefs greater than 0.5. The graph traversal may be stopped whenever an unobserved path node is reached. A single iteration of this algorithm can be shown to be $O(n^3)$, and the complexity of the entire algorithm is $O(n^5)$.

Iterative MPE in polytrees is similar to iterative belief propagation, but instead of producing marginal posterior probabilities, it produces the most probable value assignment to the belief network nodes in each iteration. This allows us to eliminate the final fault selection of the belief propagation algorithm which contributes to the complexity and is an additional source of inaccuracy. Similarly to belief updating, the MPE computation algorithm proceeds from the evidence nodes by passing *lambda* and *pi* messages along the belief network edges. We use an approximation in which the algorithm computes the MPE for every network node traversing the graph starting from the observed symptom in the breadth-first order. A single traversal is repeated for every observed symptom and at the end, the belief values are computed for all network nodes. Because of this approximation, a single iteration of the algorithm is $O(n^4)$, and the complexity of the entire algorithm is $O(n^6)$ instead of the exponential complexity for the original Bayesian algorithm.

Incremental hypothesis updating creates a set of most likely hypotheses, where each hypothesis contains at least one fault that explains one or more observed symptoms. The algorithm proceeds in an event-driven and incremental fashion and ranks hypotheses using a belief metric. When a new symptom is observed, the set of hypotheses is updated with the explanation of the new symptom. If a hypothesis is unable to explain the new symptom, it is either removed from the set or is extended by adding a fault that can explain the symptom. Faults are added using a greedy heuristic that helps to limit the complexity of the algorithm.

In order to compare the above algorithms through comprehensive experiments with a real-life application domain, we chose the data-link layer in a bridged network in which the path ambiguity is resolved using Spanning Tree Protocol. As a result, the shape of the considered graphs is reduced to trees, thus making random generation of dependencies resembling real-life scenarios easier. We used two metrics to represent the accuracy of the algorithms: *detection rate* which is the percentage of faults that occurred in the network in a given experiment that were detected by an algorithm, and *false positive rate* which is the percentage of faults proposed by an algorithm that were not occurring in the network in a considered experiment, i.e., they were false fault hypotheses.

Figure 2(a) presents the relationship between detection rate and network size. We observe that bucket tree and MPE algorithms outperform all others by a small amount, but they do not scale well to large networks. The shape of the graphs in Figure 2(a) indicates a strong dependency of the detection rate on the network size. For small (5-node) networks, the number of symptoms observed is typically small (less than 10), which in some cases is not sufficient to precisely pinpoint the actual fault. In small networks, any mistake in fault detection significantly reduces the detection rate. When the network gets larger, the number of observed symptoms increases, thereby increasing the ability to precisely detect the faults. On the other hand, as the network size grows, the multi-fault scenarios are becoming more and more frequent. In multi-fault experiments, it is rather difficult to detect all actual faults, which leads to partially correct solutions and the decreasing accuracy of the algorithms. Under these circumstances, the incremental algorithm has good detection rates even for large networks.

Figure 2(b) presents the relationship between false positive rate and the network size. For small networks, bucket tree and MPE have lower false positive rates than the other algorithms. However, as networks get bigger, the false positive rate of MPE grows sharply suggesting that it has a tendency to propose too big a set of faults as a final hypothesis than is actually needed to explain all symptoms. Once again, the incremental algorithm provides a reasonably good false positive rate for large networks. It should be noted that this algorithm is faster than all others which allowed us to test it for networks of up to 100 nodes.

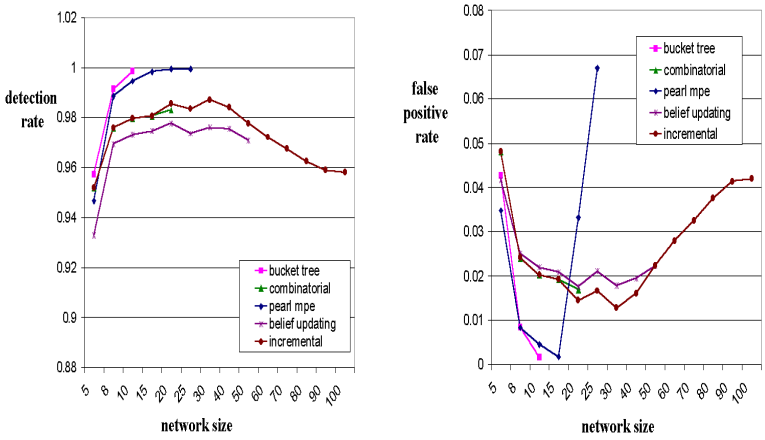

Figure 2: Comparison of accuracies achievable with various fault correlation algorithms for different network sizes: (a) detection rate, (b) false positive rate

3. SELF-HEALING AND FAULT-TOLERANCE

The necessity for self-healing mechanisms that provide service survivability (un-interrupted service) amidst random/sporadic failures in an FCS network is both critical and obvious. However, as also mentioned in the introduction, the dynamic and unpredictable nature of FCS networks coupled with the fact that FCS-network resources are scarce (hence expensive), renders the design of appropriate self-healing mechanisms extremely challenging. This section discusses a novel adaptive multi-layer, multi-service self-healing mechanism that is also sensitive to the cost, performance and complexity aspects.

A majority of the well-known self-healing mechanisms that exist to-date[2] function from a single layer [Wu92]. More specifically, they function at the physical layer (L1). Additionally, they employ the philosophy of "resource-redundancy" to handling failures whereby certain pieces of equipments are devoted solely for the purposes of restoration/failure handling. This implies the necessity for "standby" equipment that essentially remains idle during "normal" working conditions. Upon a network facility failure, an automatic switchover occurs from the malfunctioning/failed equipment (i.e., from the "working/normal" equipment) to the "dedicated/standby" equipment. Hence this self-healing mechanism is also referred to as an APS (Automatic Protection Switching) mechanism. Examples of the widely used APS self-healing mechanisms in use today are SONET self-healing rings, Bidirectional Line Switched Rings (BLSRs), etc.

While the main advantage of an APS philosophy is the excellent response time (restoration delay < ~50msecs), it has several disadvantages from the perspective of being useful in the context of FCS networks. Firstly, it is very resource expensive since by definition it works on the principle of resource redundancy. FCS networks however are severely resource-constrained. Next, it is limited to handling the class of hard failures (i.e., equipment failures) alone. Due to the random/sporadic nature of the FCS networks, a substantial amount of failures are likely to fall under the category "soft" failures, i.e., failures that result from the stochastic nature of the network (e.g., excessive performance degradation, loss of signal). Thus, while these APS-based mechanisms may have been okay for the environment they were tailored for (i.e., telecom networks), they will certainly not be suitable in their present form for the unpredictable and resource-sensitive FCS environment. Furthermore, in light of the diverse QoS and survivability requirements of FCS applications, it may even be useless to provide a uniform degree of restoration (i.e., with restoration delays of < 50 msecs) to all of the applications. For example, while mission critical and delay sensitive applications may require stringent restoration delay guarantees, loss sensitive applications, such as FCS terrain information may be okay with delayed albeit guaranteed restoration. Hence, a purely APS-based mechanism that is insensitive to the different survivability mechanisms will not suffice, and in fact may also prove to be futile in the unpredictable and random/stochastic FCS environment.

[2] Additionally, we observe that the well-known self-healing systems that are used in practice exist largely in the context of wireline (telecom) systems. Widespread and commercial deployment of self-healing in the wireless networks is still in its early stages [Kant02].

3.1 Dynamic, policy-based multi-layer multi-service self-healing mechanisms for FCS networks

In light of the above-mentioned shortcomings, we propose a new approach to providing self-healing in FCS networks, whereby we deviate from the traditionally used APS-based single layer philosophy [Kant02]. The first aspect/feature of our proposed approach is that restoration is moved "up" in the network, i.e., to the network layer (L3) and is non-APS based. Another key aspect is the use of two simultaneous layers (vs. just one layer) for providing self-healing. Based on the delay, loss and "access-to-network-resource" properties, our design will involve the use of both L3 and L1. More specifically, based on our analysis of the properties and merits of each of these layers with regards to self-healing, we design a novel multi-layer mechanism that is sensitive to the cost, performance and complexity aspects to provide service survivability in the dynamic and unpredictable FCS environment. Self-healing at L3 is provided by an on-demand trigger for re-configuration (a logical re-configuration in this case) and re-routing, whereby new routes based on the survivability requirements will be computed around the failed network element. Since failures of the network element may be either hard (e.g., router/switch failure) or soft (e.g., excessive loss on a particular link), the affected services will be restored by performing a dynamic, on-demand re-routing around the "failed" element(s) based on their survivability requirements, i.e., the high priority applications will be restored first followed by the others. In addition to L3 self-healing, we propose to tap into the excellent delay properties of L1 self-healing by triggering a simultaneous but judicious/cost-efficient L1 restoration. This is achieved by using 1:N N>>1 resource redundancy, i.e., very limited dedication of resources, which may, for example, be used only in the case of mission critical applications (which recall have highly stringent delay *and* loss requirements).

In the proposed restoration mechanism, upon receipt of a failure notification (soft and/or hard failure), the fault management (FM) subsystem will invoke the fault correlation and localization mechanisms discussed in Section 2 to locate the underlying cause of the

observed symptom. To provide self-healing, the FM subsystem at the Network Management Layer (NML) interacts with the Service Management Layer (SML) to obtain a list of the affected applications and their survivability requirements. Based on these requirements, the FM subsystem will invoke a re-routing-based-self-healing at the network layer (L3) to restore the affected applications in a prioritized manner, i.e., the high priority applications will be re-routed (hence restored) first followed by the others. More specifically, a dynamic event-related trigger will be provided by the proposed L3 self-healing mechanism to the routing mechanism to re-compute new routes. In order to arrive at an appropriate route (i.e., avoid the faulty element/s) the proposed self-healing will also provide the routing mechanism information on the "cost" of network paths. Observe that while the "cost" of network paths in the case of a hard failure are automatically registered as infinite, in the case of a soft failure, it is not so. Hence, the L3 self-healing mechanism will assign a high cost value to the paths experiencing the soft failure and provide this information to the routing procedure. The routing produced will use these link costs in its route computations, thereby arriving at new and improved routes in order to restore the affected set of applications. The self-healing mechanism will also provide the survivability requirements to re-classify the affected applications so that they may be re-routed based on their priorities (and hence survivability requirements).

Since L3 self-healing will, statistically speaking, have higher restoration delays than L1 restoration, the self-healing strategy may also, based on the applications' priority and survivability requirements, simultaneously trigger a limited L1 restoration. For example, consider the case of an FCS mission critical application with very stringent loss AND delay requirements. In this case, L3 self-healing with the highest priority to the mission critical applications may be performed. Due to the stochastic nature of the network, L3 self-healing will yield statistical guarantees. While it is indeed possible to provide stringent (<100 msec) delays to these applications via prioritized L3 self-healing, if deterministic guarantees are required and/or the set of mission critical applications is very large, then the proposed self-healing mechanism will trigger a limited L1 self-healing by automatically switching a small sub-set of mission critical applications at the physical layer. However, the L1 self-healing now has the following important caveats/deviations from the traditionally used L1 self-healing. First, we design the system to contain very limited redundancy by having 1:N N>>1, i.e., by requiring just one dedicated resource for N working resources (vs. the traditionally used 1:1; with N=1). Next, since this is a multi-layer mechanism, the self-healing mechanism within the FM sub-system will pick out only a sub-set of applications to be restored at L1 - which can be policy driven (see also discussion at the end of Section 3.2) and be made to correspond to the sub-set of mission critical applications. Further, while L1 self-healing has traditionally been the default, in our case L3 is the default. The limited L1 will only be triggered if the FM subsystem determines that the set of high-priority applications cannot all be restored with low delays at L3. Since the FM subsystem is at the NML and has a network-wide view, it will compute the statistically expected values (E[.]) of the restoration delays for the high priority applications based on its knowledge of the system "state", and use this information whether or not to invoke L1 self-healing. It will also provide restoration/re-routing priority inputs so that the high-priority applications are (re)classified appropriately during the L3 self-healing.

Note that such a judicious and novel combination of L3 and L1 self-healing has the merits of providing cost and performance sensitive restoration. The reductions in cost are obvious since L3 self-healing is essentially non APS/non-redundancy based and the limited L1 requires very little redundancy. The performance sensitivity is achieved by understanding that not all of the applications will require the same degree of survivability and hence tailoring the restoration of high-priority applications first followed by the others. This in combination with limited L1 self-healing for the mission critical applications alone will achieve the performance (low average restoration delay and loss). In fact, our prior studies on multi-layer self-healing for commercial wireline telecom networks [Hsing98] provide us with useful insights into the potential of such a mechanism. However, the random, ad-hoc and unpredictable nature of FCS networks merit further investigation into the applicability of this novel multi-layer restoration philosophy, as discussed in Section 3.2.

The complexity-sensitivity aspect of our multi-layer approach is achieved as follows. Our self-healing/service survivability mechanisms are designed to work both directly and indirectly, with a majority of existing performance management (PM) and configuration management (CM) functionalities. For example, responding to and analyzing performance-related alarms (caused by "soft failures") and responding to soft failures by re-routing essentially imply close tie-in with existing PM, CM and/or routing functionalities, albeit with some modifications. Since the design is based on the characteristics of the existing management components to a large extent, we anticipate very little added complexity to achieve the desired fault-tolerance/survivability. The self-healing strategy lends itself excellently to the rapidly emerging policy-based network management (PBNM) paradigm because the self-healing alternatives are indeed expressed as well-defined policies. Example policies include rules defining the choice of a particular restoration layer for a given set of applications, rules governing the restoration priorities

of the applications at any given layer, etc. In fact, meta-rules that check for consistency may also be defined as policies, for example, policies that prevent simultaneous restoration of a given application at more than one layer, which constitutes a very important policy set.

In light of stringent security requirements in an FCS environment, we note that our self-healing mechanism provides the added advantage of tightly integrating the self-healing with the SM sub-system. It also provides the much-needed integration of FM, CM and PM that does not exist in the current commercial systems. Such integration is indeed vital to the functioning of the FCS, since severe inconsistencies may arise otherwise. Consider for example the case when a link frequency carrying high priority/mission critical applications experience high BER. In this case a soft-failure will be registered by the PM component. Due to the urgency of the situation, the FM component will try to perform a high priority L3 self-healing. If one of the solutions is to dynamically and rapidly re-configure the primary path, then the self-healing will trigger such an action, calling for immediate co-ordination with the CM process.

Hence in such a scenario, it becomes almost imperative to design an integrated FM, CM and PM system, since to do otherwise may not only yield a bad system design but also result in severe network management discrepancies. If for example, the CM was not invoked in time, but the FM/self-healing sub-processes were to go ahead and mark different routing/configuration paths for the affected applications, there arises a serious discrepancy between the originally configured path and the newly configured path/info. This can lead to serious damage (loss of information, compromise of frequency, etc.) which can be extremely detrimental, especially when high-priority/mission critical applications are involved. Our policy-based self-healing also offers the much needed mechanism to define policies that ensure the consistency in the operations of each of the individual NM piece-parts/sub-systems.

3.2 Tactical Battlefield Network Simulation Model and Results

We have adopted the following systematic and phased approach in building Proof-of-Concept (POC) simulation models [Kelton91]. Since L3 self-healing is characterized by statistical performance and is more challenging (in terms of modeling and simulation (M&S) than L1 APS self-healing, which in-fact has deterministic performance, we have first modeled L3 self-healing for a tactical and strategic network as described below and studied its restoration performance (captured via restoration delay and loss performance statistics). Our results indicate the excellent potential of this approach. In order to factor in L1, we note that the performance of L1 self-healing is deterministic in value (50 msecs). Its cost is inversely proportional to "N". Since the cost of L3 self-healing is zero (in terms of extra resources), the total cost is essentially equal to the cost of L1, which by our very definition is very low (low resource redundancy).

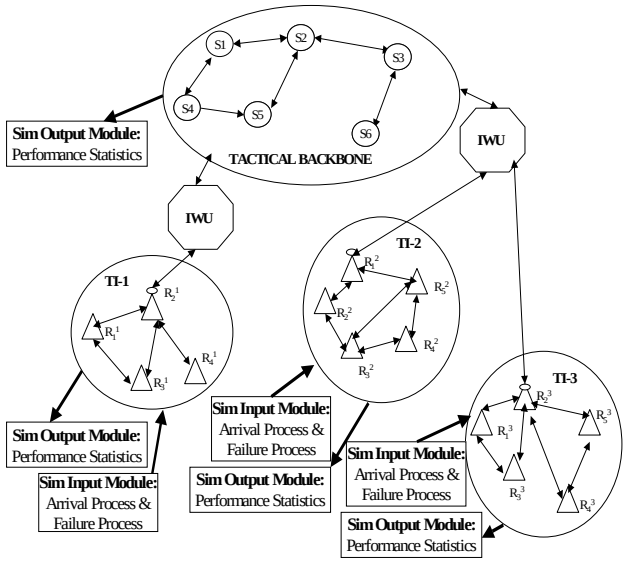

Figure 3: Tactical Battlefield Simulation Network Model

Figure 3 provides a schematic overview of the simulated POC network to illustrate and study L3 self-healing in a dynamic ad-hoc battlefield network environment. The network modeled consists of three IP based tactical internets (TI-1, TI-2 and TI-3) interconnected via a strategic ATM-based backbone (Tactical Backbone) network. Network elements (TI-routers /backbone-switches/links) can fail randomly. In the POC simulation, the various switches in the strategic backbone segment (labeled as S1 - S6) and routers in the tactical segments (labeled as R_x^y, where the subscript is the router number and superscript is the TI number that it belongs to) were made to fail randomly, to simulate an FCS environment. The block labeled IWU represents an Inter Working Unit, that will be used for interworking between the IP router-based tactical internets and the switch-based (connection-oriented) strategic ATM backbone segment. The arrivals to the network were modeled via the module labeled "arrival process", and the random failures were modeled using a variety of probabilistic distributions and labeled as "failure process" in Figure 3. L3 self-healing was designed and incorporated in the simulation model. Two sets of applications were used: (a) mission critical set and (b) non-mission critical set. The average restoration delays for class (a) and (b), were 75 msecs and 280 msecs, respectively. The restoration losses (i.e., the fraction of affected applications that were not restored) were $O(10^{-5})$ and $O(10^{-4})$, respectively, for classes (a) and (b).

Finally, before we conclude this section, we observe that our proposed self-healing mechanism offers an additionally powerful feature, namely, the capability to take *proactive* steps in case of "*impending*" soft failures. This may be achieved by simply linking the PM sub-system with the proposed self-healing mechanism in a manner such that the latter (i.e., self-healing mechanism) is made aware of any impending soft failures by checking for thresholds and "low/high-water mark" crossings. In fact, policies may be defined that look for performance abnormalities in the network, and apply rules to ensure that they are not transients and take corrective actions *before* a soft failure occurs. Once again, this can be extremely useful, especially when mission critical applications are involved.

4. CONCLUSIONS AND FUTURE WORK[3]

In this paper, we described a strategy for fault correlation and self-healing for FCS networks. We have designed and evaluated several fault localization algorithms using a layered fault propagation model transformed into a belief network. Other simulation results point to interesting properties of these algorithms that make them even more suitable to the FCS environment: resiliency to noise, ability to deal with spurious and lost symptoms, and ability to handle both positive and negative information [Steinder02b]. We are currently in the process of designing more detailed simulations of mobile battlefield wireless networks in which to test out these algorithms in order to understand the impact of mobility on the fault correlation framework. With regards to the survivability mechanisms, we described a novel adaptive policy-based multi-layer multi-service self-healing mechanism that is sensitive to the cost, performance and complexity aspects in an FCS environment. We also illustrated the excellent potential of the proposed mechanism via proof-of-concept simulations of a tactical battlefield network consisting of a strategic backbone segment interconnecting several mobile router-based tactical internet segments. Additionally, our qualitative assessment of the complexity-aspects provides valuable insights into the ease of deployment of the proposed self-healing mechanism. As continuing work, we will incorporate the deterministic and relatively easier limited-form of L1 self-healing and also perform quantitative assessment of the self-healing related interactions. The proposed self-healing mechanism also has the capability to perform *proactive* fault tolerance, an extremely useful feature in an FCS environment.

[3] The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied of the Army Research Laboratory or the U.S. Government

REFERENCES

[Gopal00] R. Gopal, "Layered model for supporting fault isolation and recovery." In *Proc. of Network Operations and Management Symposium (NOMS)*, Honolulu, HI, 2000.

[Heckerman95] D. Heckerman and M.P. Wellman, "Bayesian networks." *Communications of the ACM*, 38(3): 27-30, Mar. 1995.

[Hsing98] D.Hsing, L.Kant, and T.Wu, "Providing Service Survivability via Multi-layer Restoration Strategies in Broadband Networks." In *Proc. of Design of Reliable Communication Networks - DRCN*, Burgge, Belgium, May 17-20, 1998.

[Jakobson93] G. Jakobson and M.D. Weissman, "Alarm correlation." *IEEE Network*, 7(6): 52-59, Nov. 1993.

[Kant02] L. Kant, "Design and Performance Modeling & Simulation of Self-healing Mechanisms for Wireless Communications Networks." In *Proc. of 35th Annual Simulation Symposium*, San Francisco, April 2002.

[Katzela95] I. Katzela and M. Schwarz, "Schemes for fault identification in communication networks." *IEEE Transactions on Networking*, 3(6): 733-764, 1995.

[Kelton91] A.M.Law and W.D.Kelton, *Simulation Modeling and Analysis*, McGraw Hill Publications, 1991.

[Kliger95] S. Kliger, S. Yemini, Y. Yemini, D. Ohsie, and S. Stolfo, "A coding approach to event correlation." In [Sethi95], pp. 266-277.

[Nygate95] Y.A. Nygate, "Event correlation using rule and object based techniques." In [Sethi95], pp. 278-289.

[Pearl88] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, 1988.

[Sass02] P. Sass, "FCS communications technology for the Objective Force." 2002.

[Sethi95] A.S. Sethi, Y. Raynaud, and F. Faure-Vincent, eds. *Integrated Network Management IV*. Chapman and Hall, May 1995.

[Steinder01] M. Steinder and A.S. Sethi, "Non-deterministic diagnosis of end-to-end service failures in a multi-layer communication system." In *Proc. of ICCCN*, Scottsdale, AZ, 2001, pp. 374-379.

[Steinder02a] M. Steinder and A.S. Sethi, "End-to-end service failure diagnosis using belief networks." In *Proc. of Network Operations and Management Symposium (NOMS)*, Florence, Italy, 2002.

[Steinder02b] M. Steinder and A.S. Sethi, "Increasing robustness of fault localization through analysis of lost, spurious, and positive symptoms." In *Proc. of IEEE Infocom*, New York, NY, 2002.

[USA02] United States Army White Paper, "Concepts for the Objective Force." 2002.

[Wu92] T. Wu, "Fiber Network Service Survivability", Artech House, 1992.

[Yemini96] S.A. Yemini, S. Kliger, E. Mozes, Y. Yemini, and D. Ohsie, "High speed and robust event correlation." *IEEE Communications Magazine*, 34(5):82-90, 1996.