Diagnostics and Integration in Electric Utilities

Michael S. Moore, Ph.D., Saeed Monemi, Ph.D., Jianfeng Wang

Vanderbilt Institute for Software Integrated Systems (ISIS)
PO Box 1829, Station B
Nashville, TN 37235

James Marble, Steve Jones

Joe Wheeler Electric Membership Corporation (JWEMC)
P.O Box 460
Trinity, AL 35673

Abstract— This paper presents a new decision support system that has been developed to aid electrical utilities in diagnosing, evaluating, and planning repairs of faults during service outages. The system, called the Integrated Distribution Management System (IDMS), provides both a diagnostics component and an easily configurable integration framework. The IDMS is a cost-effective solution for fault management in small to medium sized utilities.

Index Terms— Geographic Information Systems (GIS), Supervisory Control And Data Acquisition (SCADA), Diagnostics, Integrated Distribution Management System (IDMS), Outage Restoration Management System (ORMS), Model Integrated Computing (MIC), Graphical Modeling Environment (GME)

I. Introduction

Two major concerns being faced by rural electric utilities in the current more competitive environment are 1) how to affordably and effectively manage outages in the energy distribution networks, given decreasing resources, and 2) how to cost-effectively integrate existing information systems so that they work collectively to support business activities such as diagnosis and repair planning. This paper presents a system we have developed to address these concerns for electrical utilities, called the Integrated Distribution Management System (IDMS). The IDMS integrates several information systems with a diagnostics component called the Outage Restoration Management Server (ORMS). The ORMS employs an advanced diagnostics reasoning algorithm designed specifically for electrical distribution networks to determine the location of faulty components during electrical outages, and presents the diagnostics results to the user in graphical form to aid in planning repair actions. The IDMS includes a flexible integration framework, based on Model Integrated Computing (MIC) technology that provides the integration between the ORMS and several standard Commercial Off The Shelf (COTS) systems commonly used in electrical utilities. These components include an ESRI GIS system, a Lucent Technologies Interactive Voice Response system (IVR / trouble call system), a QEI Supervisory Control And Data Acquisition system (SCADA), and a Customer Information System (CIS) database. We will show that 1) the ORMS provides accurate, relevant, and timely diagnostics results and 2) because of the use of MIC technology, the IDMS can be easily adapted to integrate other information system components used in utilities.

II. The Small Utility Environment

Small to medium sized utilities tend to have similar concerns with respect to information technology. Most utilities depend upon computer systems for managing their maps (GIS). Many have SCADA systems for remotely managing sub-stations and main switches. Most have Interactive Voice Recognition Systems (IVR) which automatically log the calls of customers reporting outages. The difficulties come when these systems have to work together, for example in the control room during an outage. A dispatcher watches the trouble call and SCADA systems, fields trouble calls, and coordinates the repair actions of linemen. The dispatcher is actually performing much of the work of integrating and fusing information together and, and manually synthesizes the solutions. It is possible to support these tasks with systems designed to perform the integration and fusion automatically. The solution synthesis can also be supported with appropriate tools. However, small utilities do not have the resources available to develop such systems internally. Hiring companies to develop custom solutions to solve these problems is extremely expensive. Not only is initial cost high, but the cost of maintaining, upgrading, and evolving custom software is out of the reach of many small utilities. The result is that many of the processes, such as fault diagnosis, repair coordination, and resource allocation are done manually by experienced staff.

The problems of improving the fault diagnosis capabilities, and in general integrating the available data systems together in support of important decision making processes need to be solved in a cost-effective and general way. Utilities need be able use and maintain these systems more independently and inexpensively, especially in light of the current more competitive environment.

We have developed a system, called the Integrated Distribution Management System, which is designed to provide diagnostics support, and eventually other decision

support tools, to small to medium sized utilities. This system addresses two of the major concerns being faced by these utilities, effective diagnostics, and maintainable and evolvable integration.

III. Integrated Distribution Management System

A. Goal

The goal of the IDMS is to provide both decision support tools and the integration required to make them work in small to medium sized utilities. One underlying requirement for the design was that the components and the integration code itself should be developed with standard, published interfaces. This approach should produce so-called *open systems*, which are much less expensive to integrate, maintain, and evolve. Toward this goal, and to avoid re-inventing the wheel, we were obliged to use as many Commercial Off the Shelf (COTS) components as possible, and concentrate on making the integration code independent of which components were chosen, so that a utility would not become locked into using a particular vendor's solutions because of large integration costs.

The first type of decision support tool we found to be most needed and relevant was a diagnostic system that would provide queues to the operators about where the faulted components are in the electrical network during outages.

The requirements of a diagnostics algorithm to be used for utilities are 1) it must function well using reasonable resources (a standard PC or workstation), it should have the ability to detect multiple non-interacting faults within a sub-station circuit within reasonable response time (10s of seconds), it should be able to take advantage of any instrumentation available (SCADA, trouble calls, etc), it should provide accurate results (few false alarms), and it should be able to pinpoint faults at any level in the network (sub-station, feeder, section, lateral, tap) in any type of component (switch, line, transformer, fuse).

We found that the algorithms currently in use were lacking in the ability to provide accurate and relevant results. One algorithm we evaluated that was in use in a small utility was a tracing algorithm designed to identify a single fault per sub-station circuit, and was not able to take advantage of SCADA measurements, which can prove

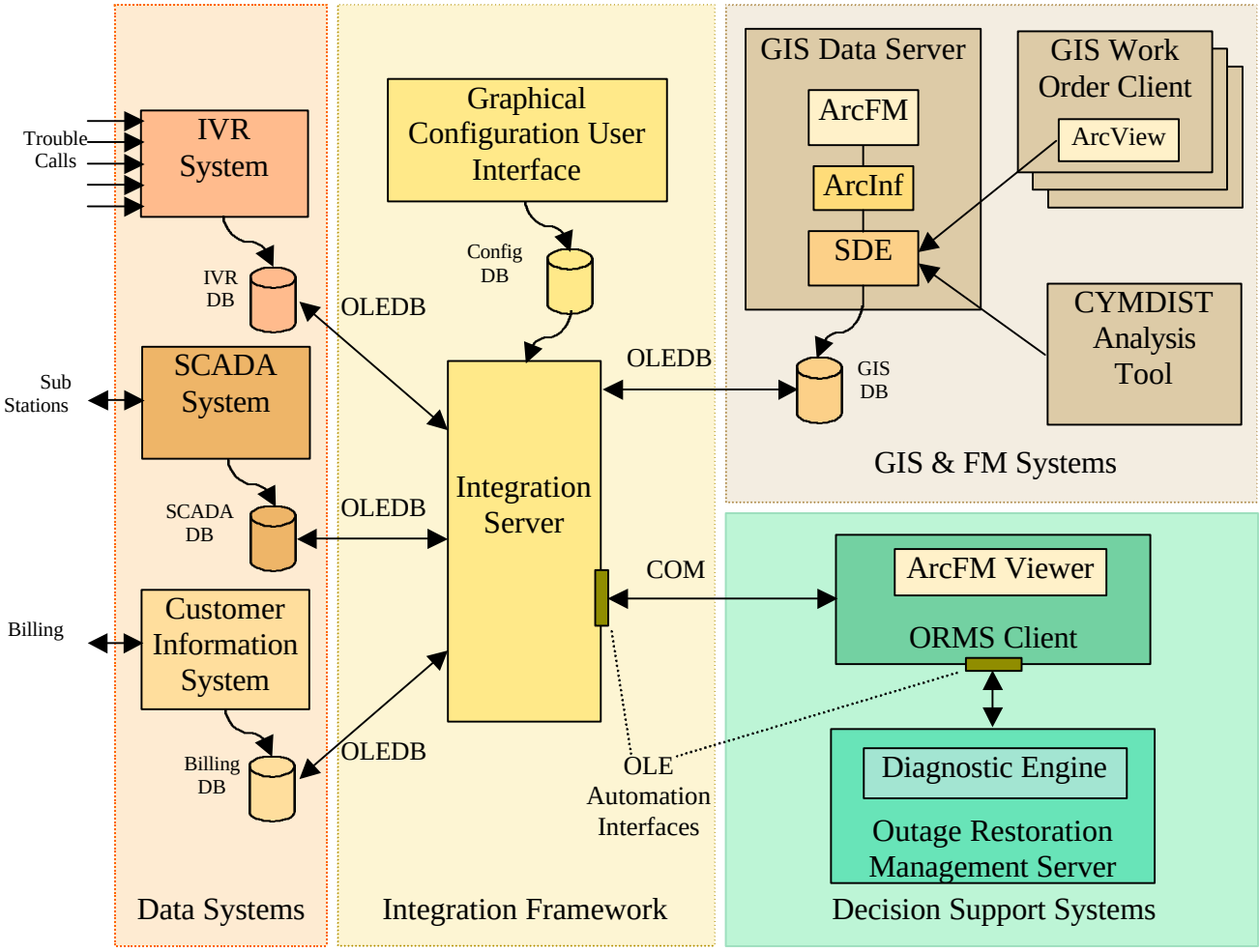

Fig. 1. The Integrated Distribution Management System

extremely useful in utility network diagnosis, as will be shown. No diagnostic systems we found would perform the functionality required with the resources available.

The IDMS was designed around the need for improved diagnostics capabilities, and the flexible, extensible integration to support it.

B. *Overall System Architecture*

The IDMS is made up of four major sub-systems, GIS & FM Systems, Data Systems, Decision Support Systems, and the Integration Framework (IF). Refer to Fig. 1.

C. *GIS & FM and Data Systems*

These systems are usually present in utilities, in varying forms. A GIS system contains a model of the circuit topology (where components are, how they are inter-connected, and some service, or customer information). Since a major goal was to promote open systems concepts, the IDMS integration framework was designed to work with GIS systems which store the circuit topology information in a standard format, such as a commercially available database (SQL Server, Oracle, SyBase, etc), or in files with either a published format or with standard access drivers available (OLEDB, ODBC, etc). Facility Management (FM) systems are used to design, maintain, control, and generally manage the network. Examples of these are work order management systems, which are used to update the GIS model as the circuit is extended and maintained, and load analysis packages such as CymDist.

The Data Systems provide additional information about the network configuration, the customers, and the health and fault status of the circuit. The status information can be thought of as instrumentation of the circuit. For example, a SCADA system will provide remote monitoring of currents, voltages, and switch positions of various remote circuit components (direct measurements). An IVR (trouble call) system will field customer phone calls and log service outages (observations of customers). A customer information database contains address and contact information of customers, service location, and billing information (additional information about the network and customers) that can be used in matching phone numbers of trouble calls to locations in the electrical network.

IV. The Outage Restoration Management Server

A. *Diagnostics*

Fault diagnosis is the process of finding the source of faulty system behavior, with a faulty behavior being a deviation from the expected system behavior. Diagnostics is a broad and varying field of study, with a large body of work. There have been many different approaches taken, including expert systems based, graph based, and model based approaches [1].

There have been several systems applied to diagnosis in electrical distribution networks, such as PDS [2], GDE [3], [4], DPNet [5], AUSTRAL, and SYDRE. However, these systems have suffered for many reasons, such as slow inference engines, limitations to known symptoms and their sources, intolerance of unreliable observations and actions, and lack of methodology for adopting a changed or similar device. The examples used to evaluate the systems have most been necessarily restricted in size and scope. Also, most systems were limited to finding a single fault to explain the observed outages.

A few diagnostics systems have been produced which are reasonable for power transmission, but there have been none capable of sufficiently diagnosing power distribution

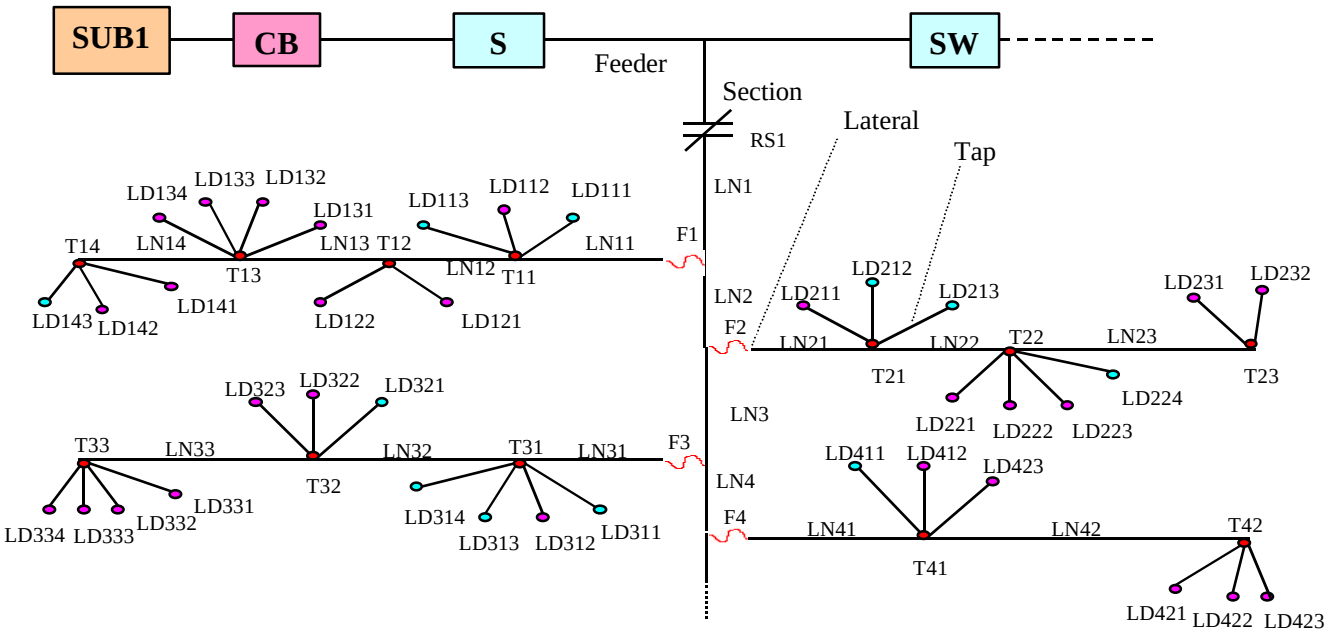

Fig. 2. EDN Structure

networks [1].

For this reason, we have developed a new diagnostics algorithm tailored specifically for diagnosing power distribution networks. It encompasses both a wealth of domain knowledge obtained through years of work within the power distribution industry and a scientific approach gained from extensive study of the general diagnostics domain. The following will explain why the diagnosis of faults in electrical distribution networks poses a unique problem, and then will present the ORMS algorithm in detail.

B. The Structure of Electrical Distribution Networks

Electrical Distribution Networks (EDN) consist of various types of electrical components, interconnected by lines. Power is input to the networks from high voltage transmission lines at *sub-stations*. The power is eventually consumed, at a lower voltage, by customers, or loads.

When viewed in this way, an electrical distribution network forms a tree topology, with the sub-stations at the roots of the tree, and the loads at the leaves. See Fig 2. Furthermore, the networks have a hierarchical structure, as will be explained in the following.

Sub-stations have several *switches* (*circuit breakers*) on their outputs, which distribute power to *feeder* lines. Feeders, or *trunk lines*, lead away from the sub-stations, and distribute power into the different service areas. Along the feeder line, there are switches that can be used to isolate parts of the circuit, but otherwise the feeders travel all the way through the service area. In some cases, the ends of feeder lines meet at switches that are nominally configured to be open, but which can be closed during outages to reconfigure the feeder circuits and temporarily restore power to areas during outages. This practice is commonly known as *back feeding*. Throughout the feeder's length are places where *section lines* branch off, normally connected by a switch. Attached to the section lines are *lateral lines*, usually with a protection device such as a fuse at the junction. Attached to the laterals are transformers, which also include protection devices, and distribute power to customers through *taps*, or *service line*s.

The entire network tree can be viewed as hierarchically composed sub-networks, as shown in Fig. 3. Note that the hierarchy levels are network, sub-stations, feeders, sections, laterals, taps, and loads.

At each hierarchy level, the circuit designers routinely place some type of protection devices attaching that level to its sub-networks. The reason for these protection devices is to provide fault isolation (i.e. to keep faults from propagating through the network). For example, if a fallen tree limb shorts a lateral line, the fuse connecting it to the section line is designed to open before the section line is damaged.

The hierarchical tree structure and the common use of protection devices are each of great importance to the design of a diagnostics algorithm. These properties allow the assumptions that 1) there will be no cycles in the fault dependency graph, and 2) faults will be localized (will not cascade). This enables us to reduce the task of diagnosing a fault to locating the area in which the fault originated, since we are assuming faults do not cascade due to the presence of correctly placed and configured protection devices.

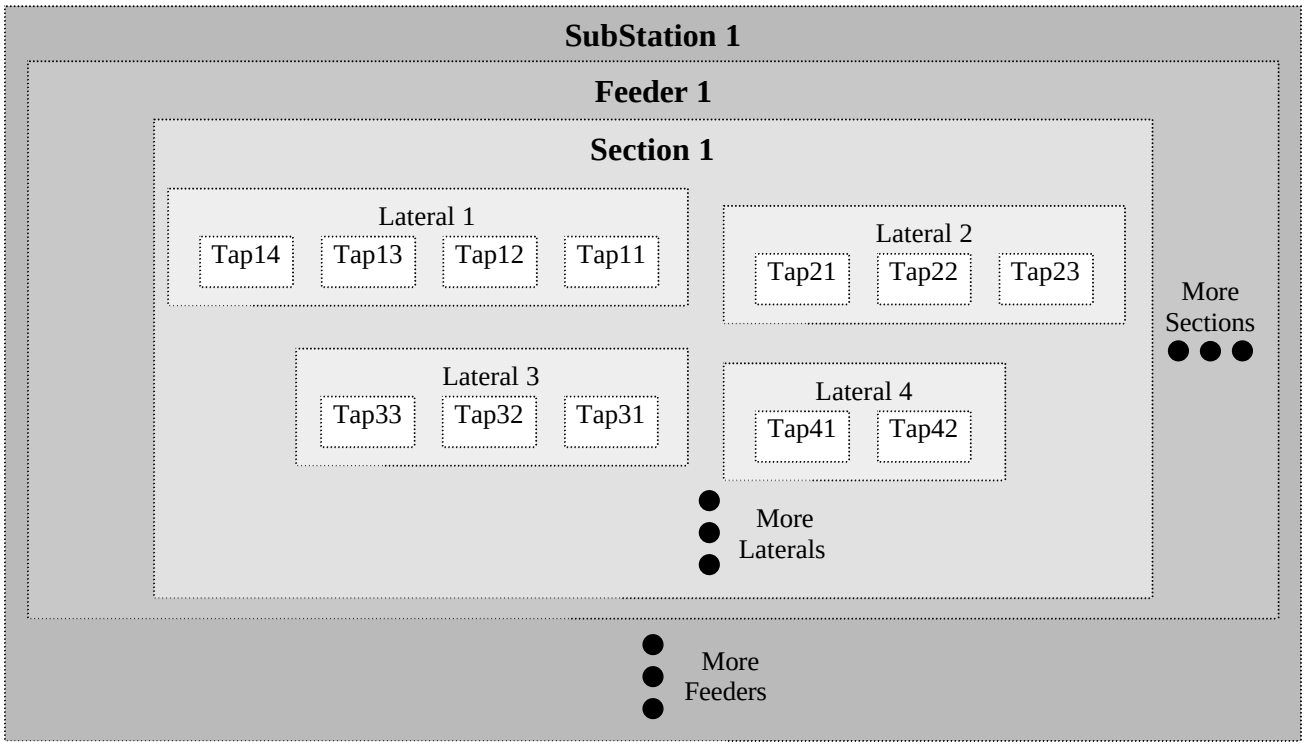

Fig. 3. EDN Hierarchy

As an example, assume several of the customers on a lateral line report an outage. A quick check determines that there have been no outage calls from any customers on the adjacent lateral lines on the same section line, so these lines are assumed to have power. This information leads to an assumption that something near the root of the first lateral line has failed. Experience says that probably the fuse is open, since it is the protection device nearest that location. Since we assume that faults do not cascade, the actual *cause* of the fuse opening was an event somewhere between the fuse and the next protection device downstream (the first transformer, in this case), probably a shorted lateral line. Thus, the event that caused the fuse to open necessarily occurred somewhere near the fuse.

Identifying that the fuse is open is sufficient information to direct a lineman to the area to quickly find the cause, repair it, and replace the fuse.

C. Instrumentation of Electrical Distribution Networks

Another unique property of EDNs poses a challenge to diagnosis of faults, and has to do with the way in which they are usually instrumented.

It is common for even small utilities to have a SCADA system, which is used to remotely monitor and control sub-station breakers, and sometimes switches in the feeder and section lines. Depending upon the amount of SCADA enabled devices in the circuit, the voltage, current, and status measurements from the SCADA system provide a certain level of instrumentation of the circuit. These devices tend to reside mainly in the sub-stations and in the feeder lines, closer to the top of the network tree.

Another more indirect form of instrumentation that is available is the observations of the customers. During outages, some customers will call the utility to report the loss of power. Most utilities have an IVR system that tracks customer calls and stores the information in a database. This information can be extremely useful in evaluating problems. We can consider these calls to be instrumentation of the circuit, but with a significant caveat. The sensors (people who could report outages) are not dependable, since not everyone calls in when his or her power is out. A customer may assume that, since his neighbor's lights are out, the entire neighborhood is probably out, so the utility must already know about it. Also, at any given time, a varying percentage of customers are not at home, especially during working hours. Thus, the information provided by the IVR system must be treated less as instrumentation of the circuit than as hints as to which parts of the circuit may be out. The trouble call data, if used as a direct indicator of which customers are out, will result in "false negative" readings (customers which are marked as OK, but which are actually out). If a customer does not call in, it is incorrect to assume that the power is on.

One conclusion that can be made from these observations is that EDNs are quite under instrumented, from the point of view of diagnostics. The dependable measurements that do exist (SCADA data) tend to be near the root of the network tree. The trouble calls provide information about the leaves of the tree, but this information is incomplete, and can result in false negative indications. The distribution of the "sensors" throughout the networks also works against us, since they are mainly clustered near the root of the tree and at the leaves, leaving out many of the hierarchy levels of the network.

D. The ORMS Diagnostics Algorithm

Because there were no diagnostics algorithms available that would produce the desired level of diagnosis with a reasonable amount of computational resources, we developed a new algorithm specifically for EDNs. The algorithm takes a novel approach toward diagnosis, in that 1) it incorporates SCADA measurements, if available, in addition to customer trouble calls 2) it exploits the hierarchical tree structure of EDNs 3) it applies domain knowledge in selecting particular components from sets suspected of being faulty 4) it uses an advanced mathematical technique based on Ordered Binary Decision Diagrams (OBDD) to deal with the combinatorial explosion in the size of the fault hypothesis space. The resulting algorithm is able to diagnose multiple, non-interacting faults in large EDNs very quickly, given reasonable resources.

The following sections will explain the ORMS diagnostics algorithm in detail.

1) Approach

An EDN will have several sub-stations, each responsible for distributing power to a service area. Given a particular switching configuration for the feeders, each sub-station represents an independent sub-circuit, from the point of view of fault management. Thus, we can reduce the task of diagnosing the entire utility network to diagnosing each of the sub-station circuits separately.

In the following discussion, we will concentrate on diagnosing starting at the sub-station, since the network faults are the composition of the sub-station level faults. Refer to Fig. 3.

Noting that the SCADA information provides measurements near the root of the network tree, that first step taken in the diagnosis algorithm should be to determine if there are any faults in SCADA enabled components. If a SCADA measurement tells us that a component is faulted, there is no us in reasoning about any trouble calls coming from customers served by that component. Thus, the algorithm first marks the customers of each faulted SCADA component as being out so that further trouble calls from that set of customers will be blocked. This is the easy step. The difficulty lies in reasoning about trouble calls that cannot be explained by a fault in a SCADA enabled component.

Consider a situation in a sub-station circuit where N customers call in and report an outage, but there are no apparent faults in the SCADA enabled components in that circuit. The simplest approach would be to trace back in the circuit from each trouble call, and select one circuit component from the set formed by taking the intersection of the sub-circuit components. This approach makes the

assumption that a single faulted component is causing all *N* trouble calls. However, in certain conditions, such as bad weather, it is possible, even likely, that the *N* trouble calls could be attributed to 2, 3, or more separate faults in the sub-station circuit. This is especially true if the trouble calls are dispersed throughout the circuit. An astute dispatcher may be able to identify multiple faults by viewing the service area map with the trouble calls highlighted. If the trouble calls show up as clustered groups on the map, the operator may assume that they are caused by separate faults, one for each cluster. With this observation, the dispatcher then ignores the single fault reported by the simple circuit-tracing algorithm, and performs a manual analysis to figure out how many faults there are, and which components are causing them.

The approach taken by the ORMS algorithm does not make the single fault assumption of simple tracing algorithms. It automates the manual analysis mentioned above, and considers the possibility that in the presence of *N* trouble calls, the number of faults could be 1, 2, *up to and including N*. The algorithm constructs fault hypothesis sets containing *N* or fewer components that 1) could completely explain the *N* trouble calls observed, and 2) contain a minimal set of components, in that if any component were removed from the set, it would no longer completely explain the observed trouble calls. The fault hypothesis sets are then ranked, and the highest-ranking set is chosen.

2) Difficulties

The difficult tasks in the fore mentioned approach are 1) finding an efficiently programmable mathematical formalism to use in generating the hypothesis sets, 2) dealing with the combinatorial size of the hypothesis space, and 3) evaluating the relative "likelihoods" of the resulting hypothesis sets.

3) ORMS Algorithm

The ORMS algorithm proceeds through the following steps to produce the fault hypothesis set:

1. Perform top down, SCADA analysis.
2. Form trouble call OBDDs.
3. Reduce the trouble call OBDDs.
4. Combine the OBDDs.
5. Perform path analysis to create hypothesis sets.
6. Evaluate the hypothesis sets.
7. Form outage groups.

Each step is explained below:

a) Top down SCADA analysis

The algorithm first traverses the network in a "top-down" fashion, starting at each sub-station. As it traverses down the network tree, if checks each SCADA enabled component for faults (if it is open, or if it has a faulted status). Upon finding a faulted component, the algorithm then creates an outage group, sets the component as the "cause" of the outage, and adds all customers dependent upon that component to the outage group. It also marks the customers as "know out", so that their trouble calls will not be processed during the more search intensive analysis of the trouble calls.

b) Trouble call OBDDs

The algorithm then analyzes the remaining trouble calls to infer the most probable set of fault components causing the outages. To do this, it traverses the network tree in a "bottom-up" fashion, beginning at each customer that called in, and ending at the sub-station. For each trouble call, it forms an OBDD.

An OBDD is a compact symbolic representation equivalent to a logical expression [6]. However, OBDDs can be manipulated symbolically (and programmatically) much more quickly than the corresponding Boolean expressions or sets, and have been shown to be useful for search space reduction in algorithms for widely varying applications [7-11].

The algorithm formulates an OBDD representing the outage status of each remaining trouble call. The OBDD is formed by traversing upwards in the network tree and logically OR-ing of the OBDD with the inverse of each component it encounters (these components are members of the customer's dependency set). The reasoning is that for a service to be out, at least one of the components it depends upon must be faulted (open). The resulting OBDD is equivalent to the Boolean logic expression

$$TC_i' = C_1' + C_2' + C_3' \ldots C_n'$$

where TC_i' means that the customer represented by TC_i is out, and Ci is a variable indicating whether component C_i is faulted or not. The OBDD corresponding to the equation above is shown in Fig. 4.

c) OBDD reduction

The algorithm them proceeds to reduce each trouble call OBDD by using domain knowledge and evaluating the OBDD at each level in the circuit hierarchy. The process assigns weights to each component based on the percentage of its customers that have called in and what type of component it is. Components that are not likely to be

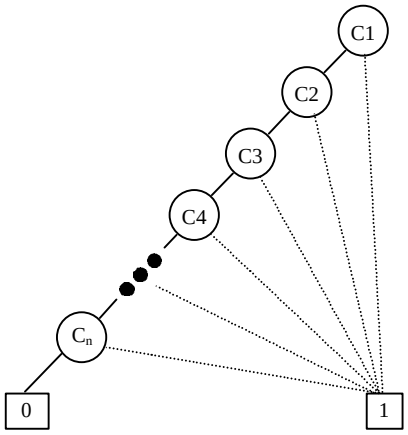

Fig. 4. OBDD for TC_i

faulted are removed from each OBDD. This is done by forcing those component variable to 1 in the OBDD.

The component weight algorithm varies depending upon the hierarchy level being analyzed, but at each level, the weight depends upon the component location (section, lateral, tap), the component responsibility (fuses protect transformers), and the number of other components downstream from the component that are out. See [1] for details about the weight assignment algorithm.

d) OBDD combination

The algorithm then combines the OBDDs by performing a logical AND between all of the reduced trouble call OBDDs from the same sub-station. This operation forms an OBDD equivalent to the logical expression TC_1' & TC_2' & TC_3' … & TC_n'. This OBDD is used to enforce the fact that each of the customers that has called in is out, but does not make assumptions about the status of the loads that did not call in.

e) Hypothesis set generation

The algorithm them creates a list of possible hypothesis sets by using an OBDD operation evaluates all paths in the OBDD that lead to "1", or "TRUE". This is equivalent to determining sets of statuses of the logical variables that will result in the OBDD evaluating to 1. Each path defines a unique hypothesis set (a set of component fault statuses which will completely and minimally explain the observed trouble calls).

f) Hypothesis set ranking

The algorithm ranks the generated hypothesis sets. For each set, it assigns a ranking equal to the product of the component weights that were generated during step 3. The reasoning in taking the product of the weights is to treat the weights roughly as probabilities (Given independent random variables a and b, P(a&b) = P(a)*P(b)) . This requires a leap of faith, since the weights are not actually probabilities. However, note that the component weight calculations take into account the percentage of that component's customers that called in, which is arguably similar to the probability that the component is itself out.

g) Outage groups

The algorithm selects the hypothesis set with the highest ranking. The selected hypothesis set is a list of components that the algorithm infers to be faulted based on the trouble calls observed. Groups of customers dependent upon each faulted component (outage groups) are formed.

4) ORMS Example

We will consider the circuit in Fig. 2 in the following example. Assume that we receive trouble calls from LD113 and LD143 only, and no SCADA faults are found. We first build the trouble call OBDDs. These are shown in Fig. 6. Note the common elements.

By applying the reduction algorithm explained in [1], we arrive at the following list of fault candidates

[F1 LN113 LN143]

with the following weights.

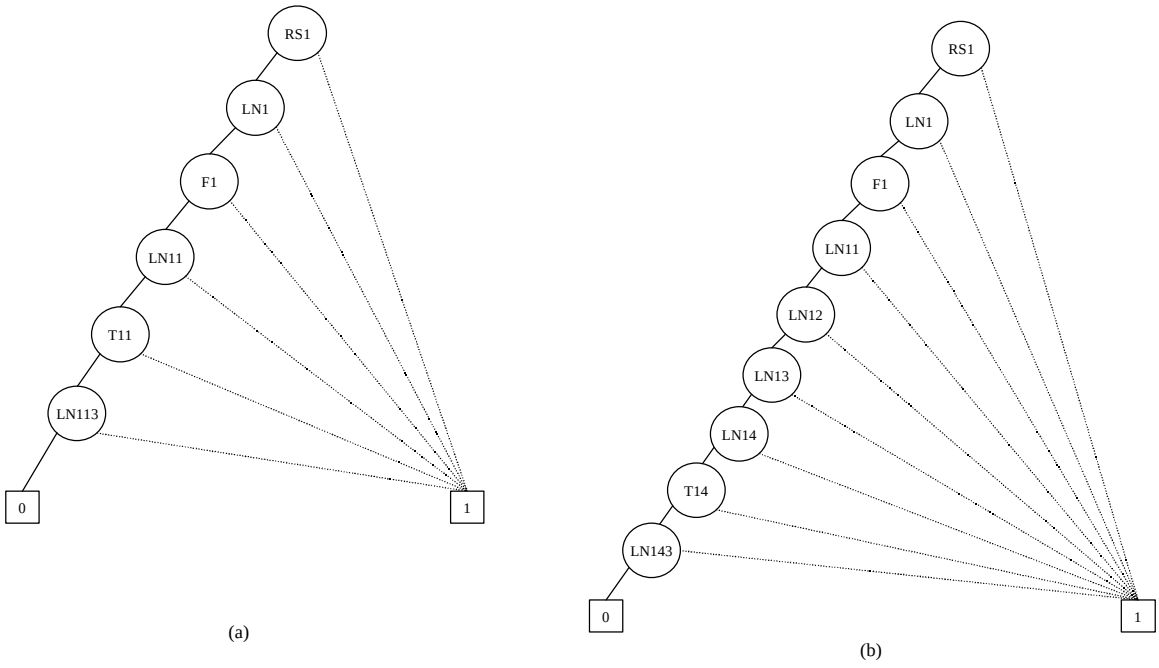

Figure 6: The OBDDs for LN113' and LN143'

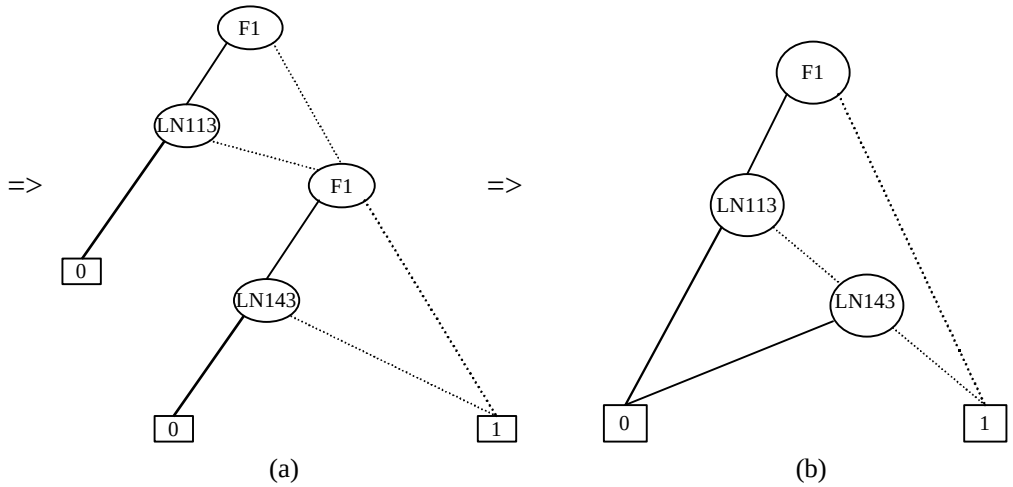

Fig. 7. Reduced composite OBDD LN113'*LN143'

[0.5 0.70 0.70]

By eliminating all nodes except for F1, LN113, and LN143 in the OBDDs, and performing the logical AND, we arrive at the reduced composite OBDD shown in Fig. 7. Generating all possible unique paths to 1 in the reduced composite OBDD, we arrive at the following hypothesis sets.

[F1]

[LN113 LN143]

Thus, the algorithm has determined that either a single component, F1, is faulted, or two components, LN113 and LN143 are both faulted. Ranking these hypotheses, we calculate the weight of the single fault hypothesis is 0.5, while the weight of the two-fault hypothesis is 0.7*0.7=0.49. Thus, the algorithm selects the single fault hypothesis, and reports that F1 is most probably faulted. Note that in this example, the ranking was very close. In such cases, the ORMS will return both hypothesis set to the dispatcher, and allow him or her to make further decisions. This approach is taken to minimize the risk of misdiagnosis.

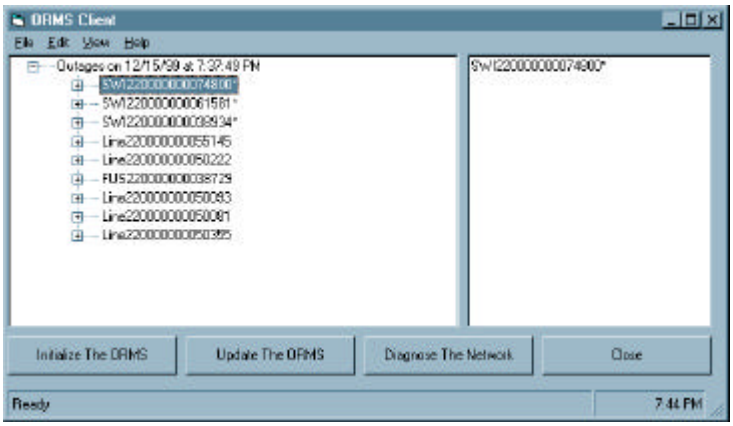

Fig. 8. The ORMS Client

E. The ORMS Client

The ORMS diagnostics algorithm is implemented by as a Microsoft OLE/DCOM component, and is controlled via a simple graphical user interface. This is called the ORMS Client, and it allows the user to initialize and update the ORMS, and run the diagnosis algorithm, and view the outages that have been diagnosed. See Fig. 8. In the left pane is a list of outages that have been diagnosed. Each outage is labeled with the Tag of the component that is at fault. Components with a "*" to their right are known to be faulted due to either SCADA measurements or trouble calls. The components without the "*" are implied to be faulted by the ORMS algorithm.

The ORMS Client is also capable of displaying the faulted components and the affected customers on the GIS map. Currently, the implementation supports this functionality only for the ESRI ArcFM GIS software. However, extending this capability is straightforward.

V. The IDMS Integration Framework

The IDMS employs Model Integrated Computing (MIC) technology, namely the Mutigraph Architecture (MGA) to create a flexible, reconfiguration, and extensible framework with which the data from the GIS system and the Data Sources are integrated with the ORMS. The Integration Framework can also provide integration for other decision support tools.

A. Model Integrated Computing

Model-Integrated Computing (MIC) is an approach to developing systems that directly addresses the problems of system integration and evolution by providing rich, domain-specific modeling environments including model analysis and model-based program synthesis tools. This technology is used to create and evolve integrated, multiple-aspect models using concepts, relations, and model composition principles routinely used in the specific field, to facilitate systems/software engineering analysis of the models, and to automatically synthesize applications from the models.

B. *The MultiGraph Architecture*

The MultiGraph Architecture (MGA) is a MIC technology that has evolved during the last decade as a software framework and infrastructure for system integration and synthesis. MGA includes generic, customizable tools for constructing domain specific modeling, analysis, and program synthesis environments. The technology has matured in major applications developed for the government and private industry, including fault detection, isolation and recovery systems for aerospace applications, on-line problem solving environments for the chemical manufacturing industry, high-performance parallel instrumentation systems, embedded simulators for turbine and rocket engine testing, and manufacturing execution systems.

C. *The GME Model Editor*

The MGA includes a configuration Graphical Model Editor (GME) that includes facilities for model building and transformation [12]. For the Integration Framework, we configured the GME to build models in a language we designed and called IDMS.

D. *The IDMS Modeling Paradigm*

The role of the IDMS models in the Integration Framework is to act as a repository for meta-information describing all available data source in the system, how they can be related, and how these meta-relations will be used by the data consumers in the system, namely the ORMS. For this reason, the IDMS modeling paradigm, or language, contains "Data Source" models, which describe all available data sources, their tables, fields, and keys, etc. Note that one assumption the Integration Framework makes about the data sources it can communicate with is that they are relational. In addition to data source models, the IDMS paradigm contains "IDMS Configuration Models", which contain "Query Models". The query models describe how the various data from the data sources can be composed to meet requests for data. See Fig. 9 for examples of data source and table. The models shown are the schema of the IDMS configuration database.

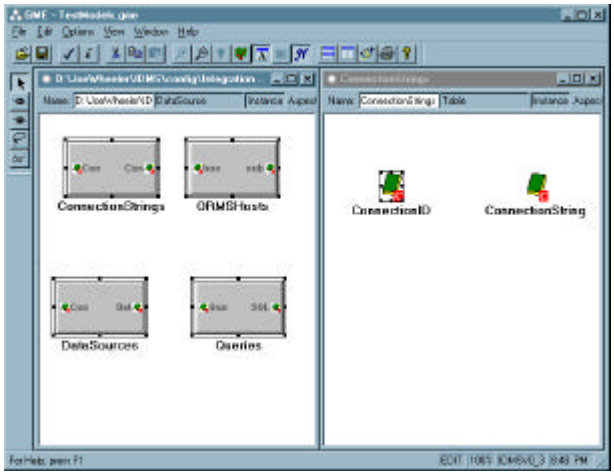

Fig. 9. Example IDMS Models

To ease the task of building the IDMS data source models, we implemented a special model interpreter that can automatically import the database schema information from any OLEDB compliant database (e.g. Oracle, SQLServer, SyBase, Access, formatted text files, and most commercial databases). This interpreter imports the tables, fields, and foreign key information. Thus, the models themselves are actually almost completely automatically generated.

E. *The IDMS Integration Engine*

The IDMS Integration Engine is the component which actually perform the requests for data, and thus provides integration between ORMS and other information system components

For instance, the ORMS client must obtain trouble calls from the Integration Framework to update the ORMS and diagnose an outage. The ORMS client calls an OLE/DCOM interface function of the Integration Engine with the identifier "GetTroubleCalls", which returns the data as an OLEDB record set object. The actual query performed by the Integration Engine will join the *outages* table from the IVR system, which includes phone numbers, with the *customers* table from the CIS, which contains transformer numbers, with the *transformers* table in the GIS database, which contains transformer identifiers and map locations. The ORMS client uses the information returned by the call to the Integration Framework to identify the customer trouble calls in the circuit. However, the ORMS client code is absolutely independent of the data location, source, or format.

F. *Model Transformation*

The IDMS models are transformed into the IDMS configuration database, which includes configuration tables read in by the Integration Engine, and other system components. When data sources change (system components are removed, replaced, or added, or data formats change) the models can be updated and the integration configuration is "resynthesized" from the models automatically.

VI. Conclusions

This paper has presented an integrated fault management system designed and developed for power distribution networks. The diagnostics component, the ORMS, is capable of efficiently identifying faulty components during power outages. The ORMS algorithm is implemented as an OLE/DCOM component, and has been integrated with several standard Commercials Off The Shelf (COTS) systems commonly used in electrical utilities. The COTS systems provide useful information to the diagnostic engine, including SCADA measurements, customer information, and customer calls. The diagnostic engine uses this data in localizing faulty components.

A. Diagnostic Capabilities

The ORMS diagnostic engine is capable of efficiently diagnosing faults in energy distribution networks. The following capabilities make this system unique:

- It can locate not only a single fault, but also multiple faults (non-interacting faults within a single feeder)
- It is able to locate and identify any faulty electrical components, including conductors.
- Because of the OBDD technique described above, the diagnosis is fast and accurate. It can process multiple outages in matter of seconds. The scalability issue is resolved by using the hierarchical structure of the feeder circuits and the fast OBDD method.
- The system is accurate and reliable because the algorithm takes advantage of all available data. It considers the component properties, their relationship and associations, their locations, and the weather conditions when ranking the fault hypothesis. It also uses historical knowledge of component failures in the decision making process. These properties make the diagnostics approach more realistic and applicable to real-world utility networks.
- Component properties, their relationship and association, and their locations are considered in component ranking.

These features have been received well among the scientists and engineers performing research in the area of diagnosis of energy distribution networks. The fact that a number of major research institutes, GIS researchers, and utility automation vendors have contacted us seeking more information about the ORMS supports this claim.

B. Integration

The use of MIC in the Integration Framework has produced a flexible, extensible, and easy to manage integration between ORMS, the data source, and the GIS system. The approach has proven to achieve almost effortless integration, and reconfiguration. Although currently the IF has been used only with the ORMS, we believe that it is capable of supporting the data integration needs of many decision support applications.

VII. Future Work

We plan to extend the IDMS to include other decision support tools. One example is a repair-planning tool, which could use GPS data to track the locations of repair crews, and efficiently schedule repairs. Another is a feeder reconfiguration support tool, which would suggest switching actions to reconfigure feeders to back-feed sections during massive outages. This tool could take advantage of load analysis software in predicting the effects of various switching plans.

In addition to new decision support tools, we plan to attempt to integrate the ORMS with GIS and mapping system from other vendors to exercise the flexibility of the Integration Framework.

VIII. Refrences

[1] S. Monemi, “Fault Management Systems in Energy Distribution Network Environments”, Ph. D. Dissertation, Vanderbilt University, Dec. 1999.

[2] R. Osborne, “Online AI-based Turbine Generator Diagnostics”, AI Magazine, pp 103, Fall, 1986.

[3] J. De Kleer, and B. C. Williams, “Diagnosing multiple faults”, Artificial Intelligence. Vol. 32, pp 97-130, 1987.

[4] P. Struss, “Model-based diagnosis-introduction and survey of recent work by ARM”, Technical Report INF 2 ARM-16-90, Siemenr, Munchen, 1990.

[5] Beschta, O. Dressler, H. Freitag, M. Montag, and P. Struss, “A model-based approach to fault localisation in power trasmission networks”, Intelligent Systems Engineering, Spring 1993.

[6] S. B. Akers, “Binary Decision Diagrams”, IEEE Trans. Comput., vol C-27, pp. 509-516, Jun 1978.

[7] R. E. Bryant, “Symbolic Boolean Manipulation with Ordered Binary Decision Diagrams”, CMU-CS-92-160, Pittsburgh, PA, Jul 1992.

[8] S. B. Akers, “Functional Testing With Binary Decision Diagrams”, General Electric Company, Electronics Laboratory, Syracuse, N.Y. pp 75-82.

[9] M. Fujita, H. Fujisawa, and N. Kawato, “Evaluation and improvements of a Boolean comparison program based on binary decision diagrams”, International Conference on Computer-Aided Design (Santa Clara, Nov.), IEEE, New York, pp 2-5, 1988.

[10] S. Malik, A. Wang, R. K. Brayton, and A. Sangiovanni-Vincentelli, “Logic verification using binary decision diagrams in a logic synthesis environments”, International Conference on Computer-Aided Design (Santa Clara, Nov), IEEE, New York, pp 6-9, 1988.

[11] J. Sztipanovts and A. Misra, “Diagnosis of Discrete Event Systems Using Ordered Binary Decision Diagrams”, Conf. Proc. 7th International workshop on principles of diagnosis (DX96), pp. 232-238 Val Morin, Quebec, Canada, October 13-16, 1996.

[12] G. Karsai and A. Ledeczi, “ A Graphical Modeling Environment for the Multigraph Architecture”, ISIS, Vanderbilt University, Manual ver. 0.5, Nashville, TN, 1998.