УДК 681.518.3

РОЛИК А.И.,
ТИМОФЕЕВА Ю.С.,
ТУРСКИЙ Н.И.

УПРАВЛЕНИЕ УСТРАНЕНИЕМ НЕИСПРАВНОСТЕЙ В ИТ-СИСТЕМАХ

Статья посвящена проблемам повышения эффективности поиска и устранения неисправностей в ИТ-системах. Предложены и проанализированы способы определения значений пороговых величин для трехпороговой схемы принятия решений. Предложен метод локализации неисправностей в ИТ-системах, объединяющий использование пассивного сбора симптомов и активных проверок. Разработана структура подсистемы управления устранением неисправностей, реализующая предложенные методы.

This article is dedicated to the problems of increasing of efficiency of fault search and elimination in IT-systems. Methods of threshold values determination for the three-threshold scheme are proposed and analyzed. Method of fault localization in IT-systems that incorporates passive symptom gathering and active probing is proposed. Fault management system which uses these methods was developed.

Введение

В настоящее время практически все организации и предприятия для автоматизации выполнения бизнес-процессов или процессов деятельности используют различные информационные технологии (ИТ), которые разворачиваются на основе ИТ-системы, включающей в себя разнообразные информационные и телекоммуникационные ресурсы. Для эффективного управления ИТ-системой и рационального использования ИТ-ресурсов разрабатываются и внедряются различные системы управления ИТ-инфраструктурой (СУИ) [1].

На СУИ, наряду с решением задач автоматизированного управления отдельными технологиями, оптимального распределения ограниченных ИТ-ресурсов [2—4], управления эксплуатацией и взаимодействием администраторов функциональных или технологических подсистем, а также решения множества других, чрезвычайно сложных и важных задач, возлагается ответственность за эффективное управление устранением неисправностей в ИТ-системе. Актуальность решения задачи восстановления работоспособности и качества функционирования ИТ-системы в кратчайшие сроки обусловлена высокой стоимостью информационных технологий, а также существенными потерями бизнеса от простоя или неэффективного использования ИТ-ресурсов, вызванных различного рода неисправностями или ненадлежащим функционированием компонентов ИТ-системы. Поэтому быстрый поиск неисправностей в ИТ-системе и проведение рациональных восстановительных мероприятий представляет собой важнейшую задачу с практической и экономической точек зрения. Кроме того, задача эффективного управления устранением неисправностей представляет большой научный интерес, который обусловлен необходимостью создания множества моделей функционирования компонентов ИТ-системы, идентификации классов и объектов управления, разработки алгоритмов обнаружения и локализации неисправностей, создания структуры подсистемы управления устранением неисправностей, а также множества других сложных и трудоемких задач. Поэтому данная работа посвящена решению вопросов, связанных с построением подсистемы управления устранением неисправностей (ПУН).

Анализ проблем управления устранением неисправностей

Все задачи управления, решаемые СУИ при устранении неисправностей в ИТ-системе, разделяются на три иерархических уровня: обнаружение, локализация и устранение неисправностей.

Для обнаружения неисправностей осуществляется сбор, обработка и анализ сведений о функционировании элементов и подсистем ИТ-системы. Для сбора информации о состоянии элементов ИТ-системы используются различные методы мониторинга и определения состояния элементов и подсистем ИТ-системы. В крупных корпорациях и организациях в ИТ-системе используется множество информационных и телекоммуникационных

технологий, а также большое количество средств вычислительной техники и коммуникационного оборудования различных производителей, которые не всегда поддерживают стандартные протоколы мониторинга. Поэтому определение состояния и качества функционирования множества элементов ИТ-системы, особенно программных, представляет существенные сложности для средств мониторинга. Для преодоления этих проблем могут использоваться программные агенты [5], устанавливаемые на элементы ИТ-системы в случаях, когда они не поддерживают стандартные протоколы мониторинга и управления, например, SNMP, CMIP и пр., когда использование стандартных протоколов запрещено из соображений безопасности или когда стандартными средствами нельзя получить всю необходимую информацию. В больших и сложных ИТ-системах выбор методов и средств мониторинга вызывает существенные затруднения даже для высококвалифицированного ИТ-подразделения.

Локализация неисправностей — один из самых сложных этапов жизненного цикла устранения проблем в ИТ-системах. Сложность поиска места возникновения неисправностей в ИТ-инфраструктуре обусловлена огромным количеством аппаратно-программных компонентов ИТ-системы и влиянием состояния одних элементов на работу других. Для решения этой проблемы используются многочисленные методы и алгоритмы, применимые только для решения отдельных задач в определенных условиях, поскольку сложная природа ИТ-инфраструктуры делает невозможным создание единого унифицированного метода или алгоритма локализации неисправностей.

Непосредственное устранение неисправностей после их точной локализации представляет собой самую простую задачу из всего спектра проблем управления устранением неисправностей. В то же время необходимость быстрого восстановления работоспособности ИТ-системы требует автоматизации процессов принятия решений администраторами и реализации возможности автоматического управления оборудованием и программным обеспечением для быстрой ликвидации неисправностей, предполагающего, например, автоматическую перезагрузку оборудования при зависании, переключение на резерв и пр. Существует ряд многофункциональных, в том числе и фирменных ПУН, которые часто входят в состав СУИ и позволяют автоматизировать ряд процессов восстановления работоспособности ИТ-систем. Однако, большое количество используемых информационных и телекоммуникационных технологий, сложность и многообразие структур и топологий корпоративных ИТ-систем, огромный спектр решаемых в системах задач, существенно отличающиеся требования к ИТ-системам со стороны бизнес-процессов и пр. делают невозможным принципиальное создание универсальной ПУН, способной решать все задачи автоматизации устранения неисправностей. Поэтому в настоящее время разрабатываются различные ПУН, предназначенные для автоматизации деятельности ИТ-подразделений. Для создания по сути специализированных ПУН необходимо разработать структуру и алгоритмы работы, определить механизмы реализации управляющих решений и решить множество других задач.

Таким образом, отсутствие хорошо проработанных принципов построения ПУН делают необходимым создание множества разнообразных моделей ИТ-систем для их эффективного мониторинга, разработки большого класса алгоритмов локализации неисправностей, методов и средств автоматизированного восстановления работоспособности ИТ-систем.

Анализ методов решения задач автоматизированного управления устранением неисправностей в ИТ-системах

Практически все бизнес-процессы крупных организаций поддерживаются информационными технологиями, поэтому незначительные нарушения в работе ИТ-системы могут вызвать простои в работе организации или ухудшить качество услуг, предоставляемых клиентам, что, как правило, влечет за собой существенные финансовые потери. Поэтому своевременная диагностика и локализация неисправностей, а также максимально быстрое восстановление работоспособности компонентов ИТ-системы является одной из важнейших задач СУИ.

Для обнаружения неисправностей определяется состояние и качество функционирования элементов ИТ-системы, после чего значения параметров работы элементов, подсистем

и пр. сравниваются с пороговыми значениями. Работа [6] посвящена анализу качества функционирования элементов информационно-телекоммуникационных систем, но в ней отсутствуют рекомендации о дифференциации состояний элементов ИТ-системы.

В [7] рассматриваются проблемы функционального мониторинга с использованием пороговых схем, однако при принятии решений участвуют только простые пороговые схемы без гистерезиса. Схемы принятия решений, предложенные в [8] и апробированные в [9], используют локальные механизмы гистерезиса, позволяющие существенно сократить количество срабатываний решающих схем, но при этом не рассматриваются вопросы, связанные с определением значений устанавливаемых порогов.

Для обнаружения неисправностей в компьютерных сетях предлагается использовать либо статистический, либо реактивный мониторинги [10]. При этом не рассматриваются вопросы использования этих видов мониторинга в тесной взаимосвязи.

В телекоммуникационных системах используется два основных метода — опрос и передача сообщений о событиях [11], а другие методы, как правило, не рассматриваются. В то же время использование только этих методов не позволяет обеспечить эффективный мониторинг ИТ-систем.

Для поиска и локализации неисправностей в телекоммуникационных системах обычно используют два подхода: пассивная диагностика [12, 13] и активные проверки [14, 15]. В первом случае происходит пассивный сбор информации о состоянии системы (например, с помощью программных агентов [5], [16]), которая затем обрабатывается для нахождения источников неисправностей. При пассивном подходе вмешательство в работу системы минимально, однако, поиск первоисточника неисправностей может занять длительное время при наличии большого процента потерянных симптомов. Активный подход подразумевает использование тестовых проверок для определения неисправностей. К недостаткам данного метода следует отнести значительное вмешательство в работу системы, а также невозможность выявления некоторых неисправностей с помощью проверок.

В [12] рассматривается пассивный метод локализации проблем в сетях с использованием матрицы вероятностей, связывающей симптомы с возможными неисправностями. Метод позволяет ранжировать по вероятности набор потенциальных неисправностей, отличается низкой вероятностью получения ложных симптомов, однако не исключает ошибочные выводы и требует большого времени анализа для крупномасштабных сетей.

В [13] описан механизм корреляции для многоуровневой диагностики неисправностей, основанный на использовании иерархической модели и позволяющий осуществлять корреляцию сообщений о неисправностях в сети и работе распределенных приложений. При этом не учитывается возможная потеря симптомов, что существенно снижает точность результатов.

Предложенный в [14] метод поиска неисправностей в сетях с помощью активных проверок, выбираемых с учетом результатов предыдущих проверок, испытывает затруднения с выявлением сбоев и аномалий при изменении параметров функционирования.

Метод генерирования набора активных транзакций для выявления источника нарушения функционирования в больших распределенных системах, предложенный в [15], сосредотачивается на поиске редких видов неисправностей.

Целью статьи является повышение эффективности устранения неисправностей в ИТ-системах, включая совершенствование методов и средств обнаружения и локализации неисправностей.

Обнаружение неисправностей в ИТ-системах

Для эффективного управления устранением неисправностей необходимо располагать достоверной информацией о функционировании элементов, подсистем и ИТ-системы в целом. Для получения этой информации осуществляется сбор и обработка данных о работе ИТ-системы с использованием методов мониторинга и анализа. Лучшим способом получения такой информации является непрерывный (или с малым периодом) контроль параметров работы компонентов ИТ-системы, при помощи которого обнаруживаются неисправности и принимаются управляющие решения о восстановлении работоспособности. Сложность современных ИТ-систем и большое количество предоставляемых ими услуг требуют кон-

троля большого числа различных параметров. Причем для передачи значений этих параметров используется та же телекоммуникационная сеть, что и для передачи информации пользователей, которая также является объектом контроля. В этом случае передача большого количества значений всевозможных параметров, необходимых для обнаружения и устранения неисправностей, создает большую нагрузку на ИТ-систему. Поэтому уменьшение объема передаваемой контрольной информации о функционировании ИТ-системы является важной задачей.

В телекоммуникационных сетях и ИТ-системах для обнаружения неисправностей используются различные методы мониторинга, из которых наибольшей популярностью пользуются статистический, реактивный и проактивный мониторинги.

При статистическом мониторинге СУИ на основе анализа поступающих к ней многочисленных исходных данных находит некоторые статистические закономерности, позволяющие предсказать тенденции в поведении компонентов ИТ-системы. В этом случае все необработанные данные должны поступить в ПУН, при этом возможности сокращения объема трафика мониторинга отсутствуют. Системы статистического мониторинга позволяют анализировать характеристики трафика и общую функциональность сети. Они хорошо подходят для анализа текущего состояния, перспектив и тенденций поведения телекоммуникационных сетей, но менее пригодны для обнаружения и локализации неисправностей в ИТ-системах.

При реактивном мониторинге СУИ получает информацию о состоянии компонентов ИТ-системы в реальном или псевдореальном времени. Это позволяет реагировать на множество аварийных ситуаций, которые могут прогрессировать в ИТ-системе. Аварийные ситуации обычно говорят о неисправности в сети или означают аномальное поведение компонентов ИТ-системы, которое может привести к неисправности. Такой вид мониторинга дает множество возможностей для поиска механизмов и алгоритмов минимизации объема контрольной информации, передаваемой по сети. Системы реактивного мониторинга позволяют определять только часть проблем в сложных ИТ-системах, испытывая затруднения при анализе функционирования сложных распределенных приложений. При этом диагностика и локализация ошибок в ИТ-системах производится после обнаружения неполадок, так же, как и определяются только проблемы, уже существующие в аппаратном или программном обеспечении.

Более совершенными являются средства проактивного мониторинга, которые не только обеспечивают дистанционный мониторинг в режиме реального времени, регулярные проверки исправности компонентов ИТ-системы, но и позволяют прогнозировать критические состояния системы и на ранней стадии генерировать предупреждения об ошибках, для того, чтобы предотвратить возникновение отказов в работе ИТ-системы. Такой мониторинг позволяет анализировать работоспособность распределенных многоуровневых приложений и пр. Главным отличием этих систем от реактивных является понимание логики распределенных приложений, а также способность предсказывать на основе анализа накопленных данных возможные сценарии развития текущей ситуации. За счет этого системы проактивного мониторинга могут выявлять и предсказать гораздо больше проблем в ИТ-системе, что позволяет устранять неполадки еще на этапе их зарождения и развития. Системы такого типа позволяют не только выявить конкретный некорректно работающий в данный момент аппаратный или программный элемент ИТ-системы, но и предсказать возможность отказа этого элемента в будущем, за счет чего обеспечивается более стабильная работа ИТ-системы и минимизируются издержки, вызванные с ее простоем. Недостатками такого рода систем являются относительная сложность в установке и настройке, высокая стоимость. Кроме того, такие системы, как правило, являются фирменными решениями, например Microsoft, которые хорошо работают в среде Windows и с продуктами Microsoft.

При мониторинге сетевых элементов используется два основных метода получения информации [11]: опрос и передача сообщений о событиях. В СУИ могут дополнительно использоваться методы мониторинга, основанные на сборе, обработке и передаче контрольной информации элементами функциональных и технологических подсистем, методы с использованием агентов СУИ [5, 16], а также методы получения информации о функ-

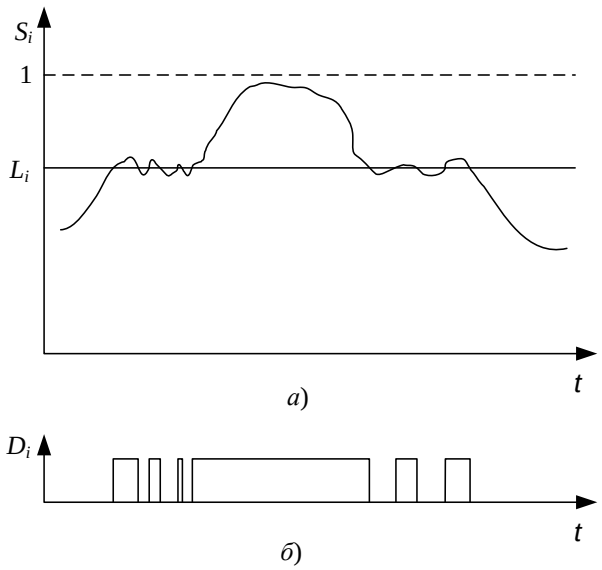
ционировании компонентов ИТ-системы по косвенным признакам. При этом могут использоваться агенты, созданные разработчиками СУИ или стандартные агенты, например, агенты SNMP, CMIP и пр.

Опрос предполагает отправку управляющей станцией запроса к элементу ИТ-системы на предоставление информации о параметрах, характеризующих его состояние. Как правило, опрос производится с фиксированным, заранее определенным периодом. Сообщения о событиях самостоятельно и асинхронно генерируются элементом ИТ-системы и передаются в ПУН. При этом используются стандартные протоколы, например, SNMP.

Сообщения о событиях вырабатываются сетевыми элементами или элементами распределенных функциональных подсистем в случае превышения заранее установленных пороговых значений.

Все типы систем мониторинга при анализе работоспособности элементов ИТ-системы производят сравнение нормированных значений параметров $S_i(t)$, где $i = 1,\ldots, I$ — количество параметров, определяющих состояние элементов, с пороговыми значениями L_i (рис. 1, а). При этом рассматривается множество параметров, характеризующих состояние элемента, либо вводится единый интегральный показатель качества функционирования простого или составного элемента, как это сделано в [6], по значению которого и оценивается состояние элемента ИТ-системы. В любом случае при пересечении величиной $S_i(t)$ порогового значения L_i изменяет значение дискретный сигнал $D_i(t)$ (рис. 1, б), получаемый на выходе компаратора, осуществляющего сравнение значений $S_i(t)$ с порогом L_i. При каждом изменении значения $D_i(t)$ в ПУН передается соответствующее сообщение.

При использовании порогов без гистерезиса (рис. 1, а) изменение $D_i(t)$ состояния $S_i(t)$ фиксируется при каждом пересечении порога L_i (рис. 1, б.). В этом случае колебания состояния в зоне порога вызывают большое количество срабатываний компаратора и значения $D_i(t)$ могут меняться очень часто. Каждое изменение состояния $S_i(t)$ вызывает необходимость передавать сведения о новом состоянии и значении параметров, приведших к изменению состояния в ПУН, с последующей активацией процедур реакции на новое состояние, а также выработкой управляющих решений, принимаемых в СУИ, и направленных на поддержание стабильной работы ИТ-системы. При этом происходит излишняя загруженность каналов связи. При использовании порогов без гистерезиса и большом количестве элементов ИТ-системы, что практически всегда имеет место в корпоративных системах, может произойти чрезмерная загруженность сети передачей служебного трафика СУИ и перегрузка программного обеспечения ПУН.

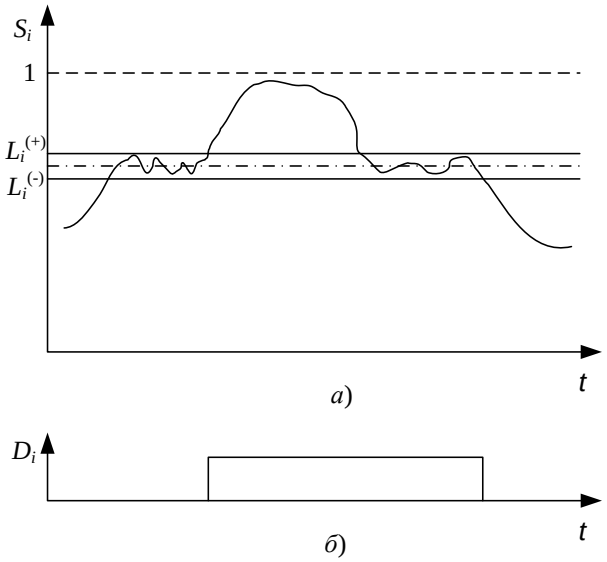


а)

б)

Рис. 1.

Поэтому в СУИ при вынесении решения о состоянии элемента ИТ-системы целесообразно использовать пороги со свойством гистерезиса (рис. 2, а), что позволяет существенно уменьшить количество срабатываний решающей схемы (рис. 2, б).



а)

б)

Рис. 2.

В этом случае значение каждого порога L_i, $i=\overline{1,I}$, преобразовывается в два пороговых значения — $L_i^{(+)}$ и $L_i^{(-)}$, а решающая схема срабатывает таким образом, что сигнал $D_i(t)$ принимает значение логической единицы при превышении значения $S_i(t)$ порога $L_i^{(+)}$ и становится равным логическому нулю, когда значение $S_i(t)$ меньше порога $L_i^{(-)}$. Если выполняется условие $L_i^{(-)} \le S_i(t) \le L_i^{(+)}$, то сигнал $D_i(t)$ сохраняет прежнее значение.

Как видно из рис. 2, б) в этом случае существенно уменьшается количество срабатываний решающей схемы определения состояния элемента ИТ-системы, поскольку решающая схема вырабатывает сигнал изменения состоянии элемента только после того, как изменение состояния надежно зафиксировано.

При использовании порогов со свойством гистерезиса (трехпороговой схемы) возникает задача определения значений $L_i^{(-)}$ и $L_i^{(+)}$.

Учитывая тот факт, что ПУН решает большое количество разнообразных задач по обнаружению и устранению множества неисправностей, происходящих в многочисленных аппаратных и программных элементах ИТ-системы и имеющих сильно различающиеся симптомы, нельзя создать универсальную методику определения значений порогов L_i, $L_i^{(-)}$ и $L_i^{(+)}$. Поэтому метод выбора значений этих порогов определяется конкретной задачей, решаемой ПУН.

Так, для оценки работоспособности распределенного приложения или функционирования подсистемы необходимо выработать критерии оценки качества работы приложения либо определить регламент функционирования подсистемы.

Для функционирования любого распределенного приложения или подсистемы необходимо обеспечить работу аппаратного и программного обеспечения рассредоточенных серверов и рабочих станций, а также сети, через которую производится взаимодействие. При этом на качество функционирования будет влиять множество различных факторов, и их комплексное влияние измерить и оценить будет чрезвычайно сложно. Поэтому для оценки качества функционирования приложения можно поступить следующим образом. На рабочих станциях периодически в автоматическом режиме запускаются контрольные задания и оценивается время их выполнения. При этом можно использовать два подхода.

В первом случае оценивается значение эталонной производительности сервера. Для этого со стороны рабочей станции, после установки операционной системы, инсталляции программного обеспечения и оптимизационной настройки вычислительной системы, с небольшим интервалом времени запускается ряд тестовых заданий. Усредненное значение результатов выполнения тестовых заданий считается эталонным и последующие выполнения контрольных заданий сравниваются с ним.

Второй подход предполагает периодическое выполнение тестовых заданий на сервере, запускаемых автоматически с удаленной рабочей станции в течение длительного времени, которое может измеряться днями или неделями. После накопления статистики с помощью методов прикладного анализа данных производится определение показателей нормальной работы приложения, определяются значения порогов L_i, $L_i^{(-)}$ и $L_i^{(+)}$, и в дальнейшем происходит сравнение результатов проверок, автоматически запускаемых с заданным интервалом, с показателями нормальной работы. Все отклонения от нормальных показателей фиксируются ПУН, после чего статистика отклонений предъявляется администратору, который принимает управляющие решения и при этом производит обучение ПУН. При ухудшении работы распределенного приложения или подсистемы администратор анализирует дополнительные признаки. Если при этом, например, произошло увеличение количества пользователей приложения, то снижение производительности — естественное явление. В этом случае администратор сбрасывает накопленную статистику о показателях нормальной работы и производится автоматическое получение новых значений показателей и порогов, либо осуществляется адаптация порогов к новым условиям работы ИТ-системы. ПУН должна игнорировать кратковременное превышение порогов, не носящее систематический характер, а также обусловленное факторами, не связанными с работой анализируемой подсистемы, а реагировать только на сбои и перегрузки компонентов подсистемы, которые проявляются в результате возникших в ней неисправностей.

Второй подход определения значений нормального режима работы представляет наибольший практический интерес для контроля

работы распределенных приложений, функциональных и технологических подсистем, поэтому ниже предлагаются способы определения значений порогов для этого метода.

Предлагается несколько вариантов определения значений порогов для трехпороговых схем принятия решений.

Все варианты используют методы прикладного анализа данных [18], накопленных в течение продолжительного интервала времени. Множество предлагаемых вариантов определения значений порогов с гистерезисом вызвано необходимостью минимизировать срабатывание решающих схем в предположении, что при дальнейшей эксплуатации подсистемы характер изменения значений контролируемых параметров не будет сильно изменяться по сравнению с характером изменения значений этих же параметров на интервале времени, когда производился сбор статистики. Срабатывание решающих схем должно происходить тогда, когда значение контролируемых параметров начинает систематически отличаться от значений, характерных для нормального режима работы, а кратковременное ухудшение показателей функционирования должно игнорироваться.

Такие требования справедливы для контроля работы ряда функциональных и технологических подсистем, но могут быть неприемлемы, например, для анализа работоспособности сетевых элементов. В последнем случае должны применяться другие способы получения значений порогов или использоваться однопороговые схемы, что и имеет место в большинстве систем сетевого мониторинга.

В качестве примера для демонстрации использован график на рис. 3 изменения значений параметра S_i, полученных в результате выполнения контрольных задач на сервере, запущенных с удаленной рабочей станции.

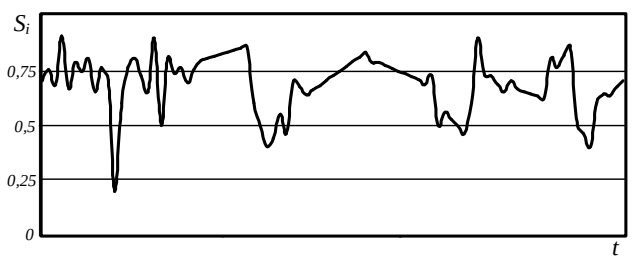


Рис. 3.

Рассмотрим три варианта построения трехпороговых схем.

В первом случае в течение продолжительного интервала времени T с периодом Δt выполняется $M = T / \Delta t$ тестовых заданий, а значение порога L_i i-го параметра определяется следующим образом:

$$L_i = \frac{\sum_{m=1}^{M} S_{i,m}}{M},$$

где M — количество тестовых заданий за время накопления статистики i-го параметра,

$S_{i,m}$, $m = \overline{1, M}$, — нормированный результат выполнения тестового задания для контроля i-го параметра в момент времени $t_m = m\Delta t$, например, нормированное время выполнения контрольного задания, запущенного на сервере с удаленной рабочей станции.

При этом в моменты времени $t_m = m\Delta t$, $m = \overline{1, M}$, кроме значения $S_{i,m}$, производится одновременная фиксация множества дополнительных признаков, например, доступность сервера, количество пользователей, обслуживаемых сервером, загруженность канала связи и пр. Эти признаки используются ПУН при локализации неисправностей в подсистеме.

Значение верхнего порога $L_i^{(+)}$ для каждого i-го параметра определяется следующим образом:

$$L_i^{(+)} = \frac{\sum_{m=1}^{M} B_{i,m}^{(+)} S_{i,m}}{\sum_{m=1}^{M} B_{i,m}^{(+)}},$$

причем

$$B_{i,m}^{(+)} = \begin{cases} 1, \text{при } S_{i,m} > L_i; \\ 0, \text{в противном случае.} \end{cases}$$

Аналогично определяется значение нижнего порога $L_i^{(-)}$ для мониторинга i-го параметра:

$$L_i^{(-)} = \frac{\sum_{m=1}^{M} B_{i,m}^{(-)} S_{i,m}}{\sum_{m=1}^{M} B_{i,m}^{(-)}},$$

здесь

$$B_{i,m}^{(-)} = \begin{cases} 1, \text{при } S_{i,m} \le L_i; \\ 0, \text{в противном случае.} \end{cases}$$

Для графика на рис. 3 нормированные значения порогов составляют: $L_i = 0,7$, $L_i^{(+)} = 0,79$,

$L_i^{(-)} = 0,49$ (рис. 4). В этом случае сигнал на выходе решающей схемы будет иметь вид, представленный на рис. 5.

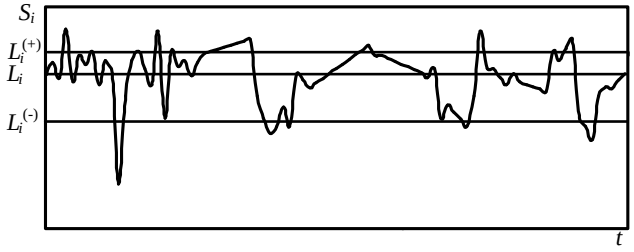


Рис. 4.

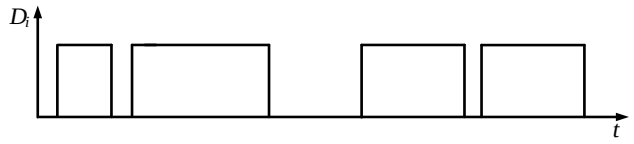


Рис. 5.

Второй способ определения значений порогов основан на аппроксимации исходных данных линейной функцией. Для этого отрезок времени, в течение которого производились замеры параметра S_i, разбивается на интервалы $\Delta T = R\Delta t$. Причем R выбирается таким образом, чтобы интервал ΔT соответствовал периоду, измеряемому десятками минут или часами и мог использоваться для отображения и учета суточной загруженности компонентов ИТ-системы. На каждом интервале ΔT определяется линейная функция, например, по методу наименьших квадратов [18], когда сначала вычисляются коэффициенты b_0 и b_1:

$$b_0 = \frac{2(2R+1)\sum_{r=1}^{R} S_{i,r} - 6\sum_{r=1}^{R} rS_{i,r}}{R(R-1)},$$

$$b_1 = \frac{12\sum_{r=1}^{R} rS_{i,r} - 6(R+1)\sum_{r=1}^{R} S_{i,r}}{\Delta tR(R-1)(R+1)},$$

где b_0 – свободный член регрессии, b_1 – угол наклона, а затем на рассматриваемом интервале строится график вида $S(t) = b_0 + b_1 t$.

Пример преобразования исходных данных (рис. 3) при разбитии отрезка времени на девять интервалов приведен на рис. 6.

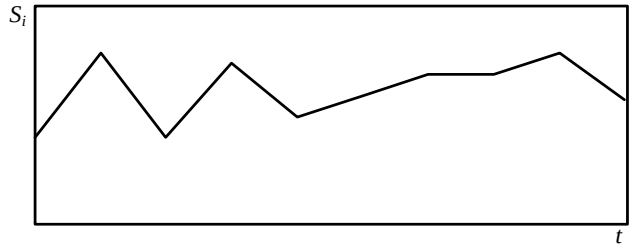


Рис. 6.

Далее производится определение порогов L_i, $L_i^{(+)}$ и $L_i^{(-)}$. Причем для вычисления значений порогов используется та же методика, что и в предыдущем способе, с той разницей, что минимальные и максимальные значения берутся из графика на рис. 6, а не из массива накопленных статистических данных о значении параметра S_i, как это имеет место в первом случае. Для графика на рис. 6 нормированные значения порогов будут $L_i = 0,58$, $L_i^{(+)} = 0,72$, $L_i^{(-)} = 0,45$ (см. рис. 7). Сигнал на выходе решающей схемы для этого случая представлен на рис. 8.

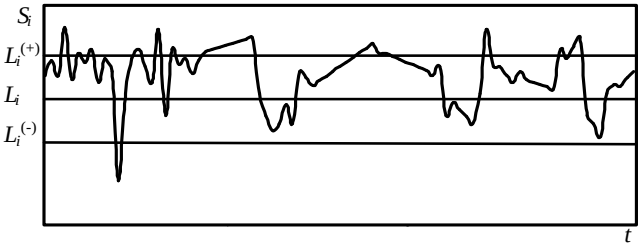


Рис. 7.

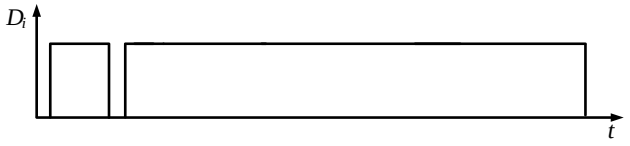


Рис. 8.

В данном случае по сравнению с предыдущим вариантом количество срабатываний решающей схемы значительно меньше, причем срабатывание произошло как реакция на явный провал кривой на рис. 7. Можно сказать, что этот метод отличается наименьшей чувствительностью.

Третий способ определения значений порогов основан на обработке накопленных данных (рис. 3) с использованием методов спектрального анализа. Для этого с помощью формул Бесселя определяются значения коэффициентов a_0, a_k и b_k:

$$a_0 = \frac{2}{M}\sum_{m=1}^{M-1} S_{i,m},$$

$$a_k = \frac{2}{M}\sum_{m=1}^{M-1} S_{i,m}\cos(km\Delta t),$$

$$b_k = \frac{2}{M}\sum_{m=1}^{M-1} S_i \sin(km\Delta t).$$

Для коэффициентов a_k и b_k, не близких к нулю, и исходных данных на рис. 3 уравнение разложения функции в ряд Фурье

$$S(t) = \frac{a_0}{2} + \sum_{k=1}^{K}(a_k\cos(km\Delta t) + b_k\sin(km\Delta t))$$

будет выглядеть следующим образом:

$$S(t) = 0,61 - 0,022\cos(6m\Delta t) + 0,0227\cos(7m\Delta t) + \\ + 0,079\sin(6m\Delta t) + 0,02\sin(8m\Delta t) + \\ + 0,036\sin(9m\Delta t),$$

а график соответствующей функции изображен на рис. 9.

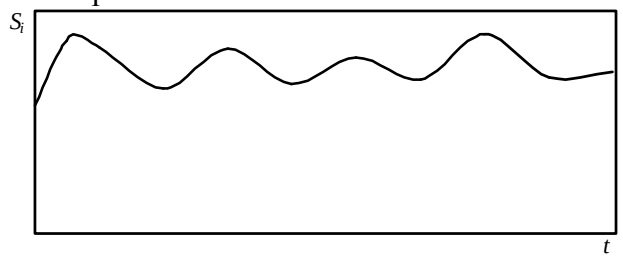


Рис. 9.

Для определения значения порогов L_i, $L_i^{(+)}$ и $L_i^{(-)}$ используется методика, описанная в первом варианте, с той разницей, что минимальные и максимальные значения берутся из графика на рис. 8. Для примера на рис. 3 нормированные значения порогов будут $L_i = 0,73$, $L_i^{(+)} = 0,81$, $L_i^{(-)} = 0,58$ (см. рис. 10), а сигнал на выходе решающей схемы для этого случая представлен на рис. 11.

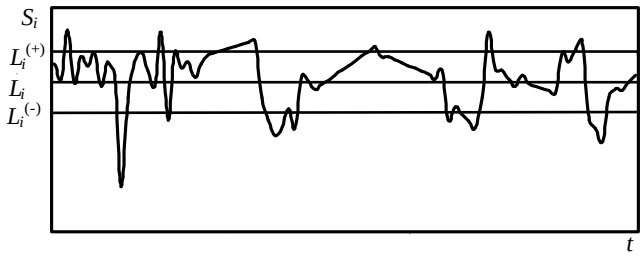


Рис. 10.

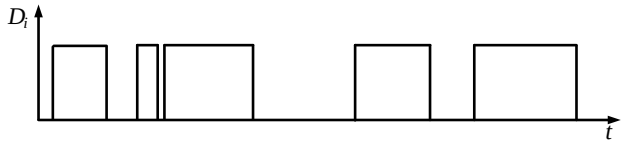


Рис. 11.

Анализируя график срабатываний решающей схемы (рис. 11), можно сказать, что данный вариант обладает наибольшей чувствительностью и позволяет выявлять практически все кратковременные отклонения от нормального режима работы.

Таким образом, для выявления большего количества отклонений от нормальной работы можно использовать первый и третий способы определения значений трехпороговых схем, для выделения только существенных и длительных отклонений от нормального режима работы — первый и второй.

Локализация неисправностей в ИТ-системах

СУИ должна обеспечивать надлежащее функционирование ИТ-системы, мониторинг и анализ работы ее составляющих, эффективное перераспределение ресурсов в случае возникновения неисправностей, диагностирование, локализацию и быстрое устранение неисправностей, а также и решение множества других задач управления ИТ-инфраструктурой. Одной из важнейших подсистем СУИ является подсистема управления неисправностями, в которой основную роль играет модуль локализации неисправностей, ответственный за выявление причины нарушения качества работы ИТ-системы, подсистем и компонентов, или полного прекращения их работы.

В [17] предложен метод локализации неисправностей в интернет-сетях и объединенных сетях, с использованием получаемых пассивно симптомов и активных проверок. На основании пассивно собранных симптомов с помощью матрицы, отображающей взаимосвязи между симптомами и неисправностями, выводятся гипотезы о наличии в сети определенных неисправностей. Проводится оценка выбранных гипотез и, если они объясняют все симптомы и не порождают симптомов, которые не наблюдаются в сети, то выдается сообщение об источниках неисправностей. В противном случае осуществляется поиск новых симптомов или обнаружение недостающих для подтверждения гипотез с помощью активных проверок. Однако в сложной ИТ-системе с большим количеством элементов сбор и обработка информации о взаимосвязях между неисправностями и их симптомами является сложной и ресурсоемкой задачей. Кроме того, в масштабах всей системы множество разных неисправностей могут вызвать одинаковые или похожие симптомы, что делает результаты локализации неисправностей менее точными и увеличивает количество ложноположительных результатов. Поэтому использование матрицы неисправность-симптом для локализации неисправностей является рациональным только для отдельных классов объектов, которые могут соответствовать как отдельным элементам системы, так и ее подсистемам, например, подсистеме документооборота, электронной почты и т. д.

В работе классификация объектов производится в зависимости от функциональности объекта, типизации его элементов и взаимосвязей внутри функциональной или технологической подсистемы. При этом учитывается степень влияния элементов и связей на качество функционирования подсистемы в целом.

Целесообразно выделение следующих классов.

Класс «Сервер — сеть — пользователи». Такому классу соответствует, например, технологическая подсистема — электронная почта. В этом случае на сервере работает приложение, которое предоставляет услуги пользователям. С точки зрения функционирования подсистемы, связь с отдельным пользователем не является критической, т. е. отсутствие этой связи еще не является неисправностью (пользователь может самостоятельно отключиться от сервера и т. д.). ПУН прежде всего должна следить за работоспособностью сервера и его доступностью по сети. Для данного класса симптомами неисправностей будут недоступность сервера или ухудшение параметров его функционирования.

Класс «Сервер – сеть – АРМ». Этому классу соответствует, например, функциональная подсистема. При этом на сервере работает приложение, которое выполняет вычисления и/или анализ на основе данных и решений, поступающих с АРМ пользователя. Связь с отдельными АРМ является критической для функционирования подсистемы. При отсутствии этой связи отсылается сообщение о неисправности. Также важной является работа сервера и отдельных АРМ. Для данного класса симптомами неисправностей будут недоступность одного из элементов подсистемы, ухудшение параметров их работы и т. д.

Класс «Сервер – сеть – сервер». Этому классу соответствует, например, взаимодействие сервера приложений с сервером баз данных, когда на сервере работает приложение, которое посылает запросы базе данных. Связь между сервером и базой данных является критической для функционирования подсистемы, так же, как и работоспособность сервера и базы данных.

Можно ввести классы, содержащие резервные сервера или другие элементы. В этом случае при недоступности или неисправности основного элемента, осуществляется автоматическое переключение на резервный. Сообщения о неисправности основного элемента не являются первоочередными, что и учитывается на этапе классификации сообщений о неисправностях по степени важности.

Предположим, что в ИТ-системе выделяется N классов. Для каждого n-го класса, $n=\overline{1,N}$, на основании анализа элементов класса и взаимосвязей между ними определяются:

– упорядоченное множество симптомов C_n т. е. различных признаков, свидетельствующих о наличии неисправностей в n-м классе, $C_n=\{c_{n,j}\}$, $j=\overline{1,J_n}$, J_n – количество симптомов для n-го класса;

– упорядоченное множество неисправностей F_n, порождающих симптомы C_n, $F_n=\{f_{n,i}\}$, $i=\overline{1,I_n}$, I_n – количество возможных неисправностей для n-го класса;

– упорядоченное множество активных проверок A_n, которые могут опровергнуть или подтвердить наличие симптомов множества C_n, $A_n=\{a_{n,k}\}$, $k=\overline{1,K_n}$, K_n – количество возможных проверок для объектов n-го класса;

– матрица размерностью $I_n\times J_n$ неисправность-симптом $Q_n=\left\|q_{n,ij}\left(c_{n,j}\mid f_{n,i}\right)\right\|$, элемент $q_{n,ij}\left(c_{n,j}\mid f_{n,i}\right)$ которой принимает значения

$$q_{n,ij}\left(c_{n,j}\mid f_{n,i}\right)=\begin{cases}1, \text{ если } j\text{-тый симптом возникает при наличии } i\text{-ой неисправности;}\\ 0, \text{ в противном случае}\end{cases}$$

– матрица размерностью $J_n\times I_n$ симптом-неисправность $P_n=\left\|p_{n,ji}\left(c_{n,j}\mid f_{n,i}\right)\right\|$, элементы которой $p_{n,ji}\left(c_{n,j}\mid f_{n,i}\right)$ соответствуют вероятности того, что j-й симптом вызван i-ой неисправностью в n-м классе и вычисляется следующим образом:

$$p_{n,ji}\left(c_{n,j}\mid f_{n,i}\right)=\frac{q_{n,ij}\left(c_{n,j}\mid f_{n,i}\right)}{\sum\limits_{f_{n,i}\in F_n} q_{n,ij}\left(c_{n,j}\mid f_{n,i}\right)};$$

– матрица размерностью $K_n\times J_n$ симптом-проверка $V_n=\left\|v_{n,kj}\left(c_{n,j},a_{n,k}\right)\right\|$, элемент которой равен нулю $v_{n,kj}\left(c_{nj},a_{nk}\right)=0$, в случае, если k-я проверка не способна обнаружить j-й симптом. В противном случае $v_{n,kj}\left(c_{n,j},a_{n,k}\right)$

определяет затраты (временные или ресурсные) на проведение *k*-ой проверки для обнаружения *j*-го симптома в *n*-м классе.

В пределах класса связи между неисправностями и симптомами определяются на основе анализа изменений значений параметров элементов класса во времени, например, с использованием методов прикладного анализа данных.

Для каждого наблюдаемого в данный момент времени симптома $c_{H,m}$ из множества $C_H = \{c_{H,m}\}$, $m = \overline{1, M_H}$, где M_H – количество наблюдаемых в настоящее время симптомов в ИТ-системе, определяется его принадлежность к одному или нескольким из *N* классам. Например, симптом «отсутствие связи с сервером X» может быть отнесен к нескольким классам, если на этом сервере работают приложения разных подсистем.

На основании матрицы неисправность-симптом P_n для каждого из этих классов выбираются те неисправности, которые объясняют наибольшее количество наблюдаемых симптомов, т. е. для каждого *n*-го класса формируется множество гипотез $H_n = \{h_{n,l}\}, l = \overline{1, L_n}$, элементами которого являются отдельные неисправности, если они могут объяснить все наблюдаемые в пределах *n*-го класса симптомы или комбинации из нескольких неисправностей. В состав гипотез включаются неисправности, имеющие максимальное значение коэффициента вклада K_B, т. е. объясняющие наибольшее количество наблюдаемых симптомов,

$$K_B\left(f_{n,i}\right) = \frac{\sum\limits_{c_{n,j} \in C_H} p_{n,ji}\left(c_{n,j} \mid f_{n,i}\right)}{\sum\limits_{c_{n,j} \in C_{f_{n,i}}} p_{n,ji}\left(c_{n,j} \mid f_{n,i}\right)},$$

где $K_B\left(f_{n,i}\right)$ – коэффициент вклада для неисправности $f_{n,i}$,

$C_{f_{n,i}}$ – множество симптомов, вызываемых неисправностью $f_{n,i}$.

Формирование множества H_n осуществляется следующим образом.

Отбор неисправностей для формирования гипотез происходит циклически. В каждом цикле неисправности с максимальным вкладом включаются во множество гипотез H_n. Симптомы, которые объясняются этими неисправностями, удаляются из множества наблюдаемых симптомов, модифицируя C_H в C'_H, а выбранные неисправности удаляются из множества F_n рассматриваемых неисправностей, модифицируя F_n в F'_n для каждого *n*-го класса. Затем поиск неисправностей с максимальным коэффициентом вклада проводится заново, при этом в качестве входных данных используются модифицированные множества C'_H и F'_n. Процедура итеративно повторяется до тех пор, пока не будут объяснены все симптомы.

Выбранные на предыдущем этапе гипотезы для каждого *n*-го класса, объединенные во множество всех гипотез $H=\{H_1, \ldots, H_n, \ldots, H_N\}$, кроме объяснения симптомов C_H, могут также порождать симптомы, о которых не поступали сообщения от подсистемы мониторинга. Существует несколько возможных причин отсутствия сообщений об этих симптомах. Данные, поступавшие от соответствующего агента мониторинга, могли потеряться, симптом не был выявлен на этапах мониторинга и анализа или предположение о наличии в системе содержащихся в гипотезах неисправностей ошибочно.

Для подтверждения или опровержения выбранных гипотез на основе матриц V_n выбираются проверки из множеств A_n, для всех $n = \overline{1, N}$. Из всех проверок, связанных с определенной гипотезой, выбираются и выполняются те, для которых значение $v_{n,kj}\left(c_{n,j}, a_{n,k}\right)$ минимально.

Если часть наблюдаемых симптомов невозможно отнести к какому-либо отдельному классу, так как они отображают взаимное влияние подсистем, принадлежащих к различным классам, то для определения источников неисправностей, вызвавших эти симптомы, необходимо провести дополнительные активные проверки.

После подтверждения гипотез входящие в них неисправности классифицируются в зависимости от степени их влияния на функционирование элементов подсистем, подсистем и ИТ-системы в целом.

При этом классификация неисправностей может производиться следующим образом:

– аварийная неисправность – неисправность, которая приводит к приостановке

функционирования элементов подсистем, подсистем или всей системы;

– функциональная неисправность – неисправность, которая обуславливает невозможность выполнения одной или нескольких функций элементов подсистем, подсистем или всей системы;

– сбой – неисправность, обусловленная однократным отказом элемента подсистемы;

– потенциальная неисправность – означает, что некоторые параметры, которые характеризуют функционирование элементов подсистем, приближаются к граничному значению, при превышении которого функционирование элемента подсистемы не будет соответствовать заданному регламенту.

Затем данные об источниках неисправностей и, при наличии, рекомендации по их устранению выдаются администратору в соответствии с установленным на этапе классификации приоритетом.

Результаты работы модуля локализации, т.е. данные про наблюдаемые симптомы и вызвавшие их источники неисправностей, заносятся в базу известных неисправностей, что облегчает дальнейшую работу ПУН.

После получения сообщений об источниках возникших в системе неисправностей и степени влияния этих неисправностей на функционирование системы, администратор, основываясь на рекомендациях ПУН, выполняет действия по устранению самостоятельно или уведомляет соответствующую службу.

Подсистема управления устранением неисправностей

Структурная схема разработанной подсистемы управления устранением неисправностей представлена на рис. 12.

ПУН состоит из модулей анализа, корреляции, локализации и связи с администратором. ПУН взаимодействует с базой данных мониторинга, базой известных неисправностей, базой данных конфигурации.

Рис. 12. Структура подсистемы управления устранением неисправностей

Модуль анализа производит предварительную обработку накопленных данных о трафике в телекоммуникационной сети и различных параметров функционирования элементов функциональных и технологических подсистем ИТ-системы. Эти данные запрашиваются модулем анализа из базы данных мониторинга, в которой содержатся результаты работы подсистемы мониторинга. В результате анализа определяются аномалии в телекоммуникационной сети и в функционировании элементов подсистем, подсистем и ИТ-системы в целом, которые не были выявлены программными агентами подсистемы мониторинга. Результаты анализа в виде сообщений о нарушении функционирования объектов (элементов подсистем, подсистем и системы в целом) передаются в модуль локализации.

Модуль корреляции уменьшает количество сообщений о неисправностях, поступивших от подсистемы мониторинга, объединяя сообщения от одного источника или имеющие одну причину и, при возможности, определяет эту причину. Выявление источника неисправности на данном этапе возможно при условии владения полной информацией о взаимосвязях элементов, от которых исходят сообщения о неисправностях. Модуль корреляции использует данные о конфигурации управляемой ИТ-системы и физических взаимосвязях ее элементов, получаемые из базы данных конфигурации. Результаты работы модуля в виде сообщений о неисправностях, т. е. симптомы неисправностей передаются модулю локализации для дальнейшего поиска источника нарушений функциональности управляемой системы.

Модуль локализации производит поиск источников неисправностей, признаками которых являются наблюдаемые симптомы. На первом этапе модуль запрашивает данные в базе известных неисправностей и проверяет, наблюдались ли подобные симптомы в системе ранее. В случае отсутствия в ней искомых данных, т. е. записей о взаимосвязях между наблюдаемыми симптомами и известными неисправностями, модуль производит поиск источников неисправностей на основании наблюдаемых симптомов. При этом задействуются данные о классификации подсистем и взаимосвязях их элементов, содержащиеся в базе данных конфигурации. Модуль находит наиболее вероятные источники неисправностей и классифицирует их в зависимости от степени влияния на функционирование элементов подсистем, подсистем и ИТ-системы в целом. Если во время поиска возникла необходимость подтвердить или опровергнуть какой-либо симптом для проверки справедливости гипотез, модуль локализации обращается с запросом к модулю активных проверок.

Модуль активных проверок подтверждает или опровергает наличие в системе симптомов, запрашиваемых модулем локализации. Происходит это с помощью выполнения активных проверок. Матрицы связи симптомов и проверок для каждого из классов хранятся в базе данных конфигурации, на основании этих матриц и происходит выбор проверок. Результаты проверок, т. е. обнаруженные симптомы или же сообщения об их отсутствии, передаются модулю локализации.

Модуль связи с администратором обеспечивает вывод информации о возникновении неисправностей, а также их причины, обнаруженные модулем локализации, что позволяет администратору изменять параметры ПУН, запрашивать и модифицировать данные в базе известных неисправностей, производить обучение ПУН и пр.

База данных мониторинга содержит информацию о значениях отслеживаемых параметров элементов и сообщения о неисправностях, т. е. о наличии симптомов. Эти данные являются результатом работы подсистемы мониторинга и собираются программными агентами с контролируемых объектов.

В базе известных неисправностей содержится информация о когда-либо обнаруженных в системе неисправностях, их симптомах и рекомендованных методах устранения этих неисправностей. После обнаружения новых неисправностей модуль локализации заносит в базу известных неисправностей данные о них: симптомы, причины и методы их устранения и пр.

Данные о конфигурации системы, о взаимосвязях элементов подсистем содержатся в базе данных конфигурации. Также в ней хранятся данные о классификации элементов и подсистем.

Выводы

В работе рассмотрены вопросы повышения эффективности устранения неисправностей в ИТ-системах. Проанализированы методы мониторинга ИТ-систем. Для выявления неисправностей в ИТ-системах с помощью порогов предложены варианты определения значений пороговых величин для трехпороговых схем.

Предложен метод локализации неисправностей в информационно-телекоммуникационной системе, объединяющий преимущества использования пассивного сбора симптомов и активных проверок, а именно: минимальное вмешательство в работу ИТ-системы и быстрое обнаружение источников неисправностей. Предложена структура подсистемы управления устранением неисправностей, которая производит обнаружение неисправностей с помощью анализа данных мониторинга, локализацию источников неисправностей, оповещает о них администратора и выдает рекомендации по устранению неисправностей. ПУН производит быстрый и достаточно точный поиск источников неисправностей в ИТ-системах. В дальнейшем необходимо ввести учет оценки степени влияния неисправностей на функционирование ИТ-системы.

Список литературы

1. Теленик С.Ф., Ролік О.І., Букасов М.М., Соколовський Р.Л. Система управління інформаційно-телекомунікаційною системою корпоративної АСУ// Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка. —2006. — № 45. — С. 112—126.

2. Ролик А.И. Модель управления перераспределением ресурсов информационно-телекоммуникАционной системы при изменении значимости бизнес-процессов// Автоматика. Автоматизация. Электротехнические комплексы и системы. ХГТУ. — 2007. — № 2 (20). — С. 73—82.
3. Теленик С.Ф., Ролік О.І., Букасов М.М. Моделі управління розподілом обмежених ресурсів в інформаційно-телекомунікаційній мережі АСУ// Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка. — 2006. — № 44. — С. 234—239.
4. Теленик С.Ф., Ролік О.І., Букасов М.М., Терещенко П.І. Управління доступом до обмежених ресурсів інформаційно-телекомунікаційної мережі АСУ військового призначення// Сб. наук. праць ЦНДІ Збройних Сил України. — 2006. — №3 (37). — С. 33—43.
5. Ролик А.И., Соколовский Р.Л. Распределение мобильных компонентов системы управления информационно-телекоммуникационной системой// Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка. — 2007. — № 47. — С. 113—124.
6. Ролик А.И., Глушко Е.В. Анализ качества функционирования элементов информационно-телекоммуникационных систем// Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка. — 2008. — № 48. — С. 113—120.
7. Cormode G., Muthukrishnan S., Yi K. Algorithms for Distributed Functional Monitoring// Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms. — San Francisco, California. — 2008. — P. 1076—1085.
8. Wuhib F., Dam M., Stadler R., Clemm A. Decentralized computation of threshold crossing alerts// Proc. 16th IEEE/IFIP International Workshop on Distributed Systems. — Barcelona, Spain. — 2005. — Vol. 3775. — P. 220—232.
9. Wuhib F., Stadler R., Clemm C. Decentralized service-level monitoring using network threshold crossing alerts// IEEE Communications Magazine. — 2006. — Vol. 44. — № 10. — P. 70—76.
10. Dilman M., Raz D. Efficient reactive monitoring// IEEE JSAC. — 2002.— Vol. 20, № 4. — P. 668—676.
11. Stallings W. SNMP, SNMPv2, SNMPv3, RMON1 and 2. — 3rd edition. AdisonWesley. — 1998. — 640 p.
12. Steinder M., Sethi A. S. Probabilistic Fault Diagnosis in Communication Systems Through Incremental Hypothesis Updating// Computer Networks.— July 2004.— vol. 45.— no. 4.— pp. 537—562.
13. Appleby K., Goldszmidt G., Steinder M. Yemanja – A Layered Event Correlation Engine for Multi-domain Server Farms// Integrated Network Management Proceedings, 2001 IEEE/IFIP International Symposium on. — 2001.— pp. 329—344.
14. Rish I., Brodie M., Odintsova N., Ma S., Grabarnik G. Real-time Problem Determination in Distributed Systems using Active Probing// Network Operations and Management Symposium. NOMS 2004. IEEE/IFIP.— April 2004.— Vol. 1.— pp. 133—146.
15. Guo J., Kar G., Kermani P. Approaches to Building Self Healing System using Dependency Analysis// Network Operations and Management Symposium. NOMS 2004. IEEE/IFIP.— April 2004.— Vol. 1.— pp. 119—132.
16. Теленик С.Ф., Ролік О.І., Терещенко П.І., Літвінцов О.В. Модель управління розподілом ресурсів інформаційно-телекомунікаційної системи збройних сил України// Зб. наук. праць ННДЦ оборонних технологій і воєнної безпеки України. — 2006.— №5 (34).— С. 117—124.
17. Tang Y., Al-Shaer E. S. Boutaba R. Active Integrated Fault Localization in Communication Networks// Integrated Network Management Proceedings. IM’2005. IEEE/IFIP International Symposium on.— May 2005.— pp. 543—556.
18. Бендат Дж., Пирсол А. Прикладной анализ случайных данных. — М.: «Мир».— 1989. — 526 с.