

УДК 658.012; 681.3.06; 621.396.6.001.66(075); 621.001.2(031)
ББК 2+3
Н 34

Научно-техническая конференция студентов, аспирантов и молодых специалистов МИЭМ, посвященная 50-летию МИЭМ. Тезисы докладов. - М. ~: МИЭМ, 2012. - 445.

ISBN 978-5-94506-314-3

В сборнике представлены тезисы докладов студентов, аспирантов и молодых специалистов.

Структура сборника включает разделы соответствующих научных направлений: прикладная математика; информационно-коммуникационные технологии; автоматизация проектирования, банки данных и знаний, интеллектуальные системы; компьютерные образовательные продукты; информационная безопасность; электроника и приборостроение; производственные технологии, нанотехнологии и новые материалы; гуманитарные и экономические науки, web-технологии и компьютерный дизайн.

Сборник тезисов представляет интерес для преподавателей, студентов, научных работников и специалистов, специализирующихся в области информационно-коммуникационных технологий, электроники, прикладной математики и экономике.

Редакционная коллегия: Азаров В.Н., Карасев М.В., Кечиев Л.Н., Львов Б.Г.,
Леохин Ю.Л., Никольский С.Н., Смирнов И.С., Титкова Н.С.,
Четвериков В.М.

Издание осуществлено с авторских оригиналов.

ISBN 978-5-94506-314-3

ББК 2+3

**© Московский государственный институт
электроники и математики
(технический университет), 2012 г.**
© Авторы, 2012г.

текстуру. Особое внимание при этом необходимо уделять типам связей между ее отдельными программными модулями. Принято различать системы следующих типов: сильносвязанные, слабосвязанные и смешанного типов. Сами связи между программными модулями могут являться как статическими (заданными начальной конфигурацией), так и динамическими (формируемыми в процессе функционирования в зависимости от выполнения определенных условий). Последние могут носить вероятностный случайный характер, что затрудняет учет их влияния на качество работы РПС.

Исходными данными для анализа надежности РПС являются статистические данные о работе ее сетевых приложений. Помимо необходимости организации корректного процесса сбора этих данных, их полноты и непротиворечивости, необходимо также обеспечить их долговременное хранение. При этом требуется оценка необходимых ресурсов (например, размер БД, полоса пропускания сетевого канала, вычислительные мощности серверов). Все это позволяет сделать вывод о тесной взаимосвязи между надежностью анализируемой РПС и качеством процесса сбора сетевых данных в ней. Это достижимо только в рамках разработки общего комплексного подхода к соответствующей проблеме.

При этом первоочередной задачей при оценке качества РПС является оценка степени взаимного влияния программных модулей в его сетевых приложениях. Получение интегрального показателя надежности должно в обязательном порядке включать соответствующий этап анализа.

Анализ надежности сетевых приложений в РПС, как уже отмечалось ранее, представляет собой решение комплекса связанных между собой проблем. Получение интегрального показателя надежности требует разработки обобщенного метода, формализующего порядок сбора необходимой входной информации, ее представление для математической модели, применение данной модели и корректная интерпретация полученных результатов. Метод должен быть достаточно универсальным и допускать адаптивную подстройку к произвольной конфигурации сетевых приложений в РПС, поскольку надежность программного модуля в сетевом приложении может значительно варьироваться в зависимости от его места в конфигурации системы, а также от истории предыдущих вызовов как данного, так и других, сопряженных с ним модулей.

В силу ограниченной применимости существующих моделей надежности для оценки надежности сетевых приложений в РПС в рамках предложенного метода необходимо разработать математическую модель, позволяющую учитывать статистические результаты автономной отладки программных модулей и использование их при получении интегрального показателя надежности сетевого приложения. Модель должна позволять не только получать показатель надежности, но и делать процесс его получения предсказуемым и экономически более эффективным.

Перечислим основные требования, которым должен удовлетворять метод обобщенной оценки надежности сетевого приложения РПС, а также требования к условиям его применения. Данный метод должен обеспечивать:

- адекватность процессам, происходящим при решении сетевых задач в реальных РПС;
- использование результатов автономной отладки программных модулей при оценке показателя надежности РПС;
- учет взаимного влияния программных модулей при их совместной работе в составе сетевого приложения;
- возможность использования любой из существующих моделей надежности автономных программных модулей в общей методике расчета надежности;

- возможность автоматизации получения оценки надежности РПС с применением программных реализаций для ЭВМ.

Литература:

1. Черкесов Г. Н. 448 Надежность аппаратно-программных комплексов. Учебное пособие. — СПб.: Питер, 2005. — 479 с: ил
2. Кузнецов М.В. 162 Методы и Алгоритмы расчета загрузки телекоммуникационной сети служебным трафиком сетевых приложений. Москва 2009.

ПРОТОТИП ПРОГРАММНО-АППАРАТНОГО КОМПЛЕКСА ДЛЯ СБОРА ДАННЫХ ОБ УРОВНЯХ СИГНАЛА GSM НА МЕСТНОСТИ

М.М. Ковалев

факультет Автоматики и вычислительной техники

Задача сбора данных об уровнях сигналов базовых станций сетей GSM, принимаемых мобильными устройствами, в сочетании с данными о том, в точке с какими географическими координатами было осуществлено измерение, возникает при разработке систем позиционирования, основывающихся только на данных о сигналах от наземных передатчиков, без использования спутниковых. Поскольку в городских условиях из-за застройки и ландшафта затухание сигнала может существенным образом отличаться от обратного квадрату расстояния, аналитического решения задачи с использованием априорных данных о расположении базовых станций не существует. Всякое решение в данном случае является приближённым и основывается на интерполяции данных фактических замеров принимаемого уровня сигнала.

Рассматриваемая в данной работе задача является частью проекта создания системы позиционирования наземного транспорта по сигналу от станций GSM, ориентированной на устройства, прикреплённые к транспортным средствам, движущимся по заранее известным маршрутам, и призванной за счёт этого превысить точность, дающуюся системами позиционирования от компаний Яндекс и Google, решающими более общую задачу. Конечной целью является создание промышленных образцов устройств, устанавливаемых на транспортные средства, а целью данной работы — создание прототипа подобной системы, способного работать как в режиме сбора данных об уровнях сигнала, связанных с данными спутникового позиционирования, так и в режиме тестирования, позволяющем сравнить данные позиционирования, даваемые разрабатываемой системой, с данными GPS.

Представленные на рынке модули GPS и GSM в значительной мере отличаются между собой как по своим возможностям и интерфейсу, так и по цене, а потому выбор именно этого узла системы осуществлялся с прицелом на его использование и в серийном образце устройства, а не только прототипе, и именно он определяет требования к остальным узлам системы. Функциональные требования, предъявляемые к нему минимальны: способность отдавать данные не только об интегральной характеристике качества сигнала, но и об уровне приёма каждой базовой станции по отдельности, в сочетании с её уникальным номером Cell ID, а также поддержка передачи данных через сети GPRS/EDGE/3G. При этом предъявляется требование минимально возможной цены устройства. Анализ рынка показал, что большинство представленных устройств облаивают существенно избыточным по отношению к данным требованиям функционалом, при этом минимальном удов-

лективющим им устройством является модуль SIM300, имеющий значительно более низкую цену, чем остальные представленные. Для прототипа был выбран модуль SIM508, представляющий собой модификацию SIM300 с добавлением приёмника сигнала GPS и имеющий аналогичный интерфейс, позволяющий, в случае использования только функций GSM, переключаться на использование SIM300 без каких-либо изменений со стороны остальных узлов. Взаимодействие с остальными узлами системы эти-ми модулями осуществляется преимущественно через интерфейс RS-232, однако включение и выключение осуществляется кратковременной подачей напряжения на выводы, предназначенные для подключения кнопки.

В связи с тем, что прототип предназначен для проведения экспериментов, в том числе связанных со значительными изменениями в алгоритме работы, было признано целесообразным использование в прототипе ОС GNU/Linux, предоставляющую целый ряд абстракций как в области взаимодействия с аппаратными средствами, так и в алгоритме работы программной части. Это является источником требований к аппаратной части: поддерживающий ядром Linux 32-разрядный процессор, не менее 8 мегабайт оперативной памяти и 50 — энергонезависимой, доступной на чтение и запись.

На основании этих требований, из решений, представленных на российском рынке, была выбрана плата SK-AT91SAM9260 от компании Starterkit, имеющая следующие характеристики: контроллер Atmel AT91SAM9260 (архитектура ARM-9), 256 мегабайт NAND Flash, 64 мегабайта SDRAM, возможность подключения к ПК по интерфейсам RS-232, JTAG, Ethernet, USB. Также, на плате уже имеется интегрированный разъём для подключения модулей SIM508 и SIM300, интерфейсы RS-232 которого подключены к портам UART контроллера, а контакты кнопки — к портам ввода-вывода. Характеристики данной платы позволяют использовать прототипы программной части, написанные на языке высокого уровня Python, поскольку её ресурсов достаточно для стабильной работы интерпретатора.

С отладочной платой поставляется комплект открыто-го ПО для первичного её запуска. Поскольку ни один из существующих загрузчиков не может полностью осуществить загрузку от момента подачи питания до передачи управления ядру Linux, используется многоуровневая схема.

В момент подачи питания, контроллер начинает исполнять программу, находящуюся в его внутренней энергоне-зависимой памяти объёмом 32 килобайта с использованием внутренней оперативной памяти объёмом 8 килобайт. Изменение содержимого этой памяти возможно только по интерфейсу JTAG. В ней находится загрузчик ROMboot, осуществляющий первоначальную инициализацию контроллера и NAND Flash, после чего передающий управле-ния второму загрузчику, находящемуся по заранее известному адресу в NAND Flash. В случае, если плата подключена к ПК по интерфейсу USB, включается альтернативная ветвь загрузки, позволяющая с помощью специального ПО записывать данные на Flash с ПК.

Стандартной практикой для загрузки Linux на ARM системах является применение загрузчика u-boot, но он не способен работать с таким малым количеством оперативной памяти, которое имеется в контроллере. Поэтому в качестве загрузчика второго уровня используется bootstrap, единственным назначением которого является инициализация основной оперативной памяти платы и передача управления загрузчику u-boot, также находящемуся по за- ранее известному адресу.

Загрузчик u-boot уже обладает широким функционалом, и помимо загрузки ядра Linux, способен также, в том числе, осуществлять запись этого ядра в определённую область Flash по протоколу TFTP через интерфейс Ethernet.

В варианте, поставляемом производителем, используется ядро Linux с минимальным окружением GNU, корневая файловая система которого полностью помещается в образ, соединённый с ядром и копируемый им при загрузке в ramfs, откуда и ведётся дальнейшая работа. Размеры этого образа ограничены тринадцатью мегабайтами, поскольку остальная часть NAND Flash используется как раздел с файловой системой ext2. Одним из вариантов решения проблемы могло бы быть изменение размера этого раздела, но это повлекло бы за собой новые проблемы со слишком большим количеством оперативной памяти, которая бы потребовалась для ramfs, кроме того, все изменения, осуществляемые в корневой файловой системе, за исключением точки монтирования раздела, не сохранялись бы при отключении питания. Достоинством такого решения является тот факт, что при сбое питания, даже если какой-то из системных файлов был при этом открыт на запись, при восстановлении питания система восстановилась из неизменного образа. Но крайне серьёзным недос-татком для целей отладки является то, что любые измене-ния в корневой системе приходится осуществлять путём пересборки образа ramfs на ПК и загрузке на плату через TFTP. Некоторые малые изменения можно осуществлять путём создания скриптов, сохраняющих при штатном вы-ключении системы разницу между текущим состоянием системы и исходным образом на раздел, однако это не сработает если изменения будут касаться файлов, обраще-ние к которым при загрузке происходит раньше, чем сра-батывает скрипт восстановления сохранённых изменений.

Архитектурно правильным решением в данном случае является перенос корневой файловой системы непосредствен-но на имеющийся раздел ext2, отказавшись от исполь-зования ramfs. Определённые трудности в таком варианте исполь-зования возникают из-за того, что u-boot предоставляет средства только для работы с неразмеченной обла-стью Flash, а раздел представляется просто занятым ме-стом, а не файловой системой. Также, он не способен пе-редавать ядру никакие аргументы или осуществлять выбор между различными ядрами, подлежащими загрузке.

Было выработано следующее решение: для первичной распаковки образа корневой файловой системы использует-ся предоставляемый производителем образ-дистрибутив, средствами которого осуществляется копирование и рас-паковка tar-архива с образом с подключённой к плате карте памяти SD на раздел во Flash. Затем, после перезагрузки системы, средствами u-boot образ ядра заменяется на новый, при сборке которого указаны параметры по умол-чанию, в том числе и задающие местонахождение корне-вой файловой системы. Если в дальнейшем требуется осу-ществить глобальные изменения, которые невозможно осу-ществить при работающей системе, средствами u-boot восстанавливается ядро производителя, работающее с ramfs, а не с раздела, а дальний процесс осуществляется с вышеописанным образом.

Для сборки собственного образа корневой файловой системы, содержащего необходимое ПО (в том числе ин-терпретатор Python, утилиты для настройки и диагностики сете-вого подключения, сервер ssh, утилиты для работы с интерфейсом RS-232 и так далее) использовался набор скриптов buildroot, позволяющий скомпилировать необхо-димое ПО под требуемую архитектуру и сформировать из него образ.

Некоторые изменения осуществлялись уже на готовом образе при работающем основном ядре, в том числе осу-

ществлена конфигурация makedev для организации доступа к портам ввода-вывода контроллера, подключённым к выводам для кнопки на модуле SIM508, как с символьному устройству, в которое можно выводить команды включения/выключения высокого уровня, через систему GPIO.

Тестирование показало, что программы на языке Python успешно запускаются и работают, а использование средствами GNU/Linux интерфейсов RS-232, подключённых к модулю SIM508 как файлов позволяет простым и прозрачным образом писать программы, осуществляющие управление модулем и обмен данными с ним с помощью AT команд. Успешно были получены как данные о географической широте и долготе средствами GPS приёмника, так и данные о том, какие базовые станции находятся в области видимости (получены их Cell ID) и каков уровень сигнала, принимаемый от каждой из них.

Таким образом, показана совместимость выбранных аппаратных средств между собой и с использованными программными, создан образ системы GNU/Linux, подходящий для решения поставленных задач и показана его работоспособность. Разработанный прототип следует считать пригодным для дальнейшего использования в рамках проекта по созданию системы позиционирования наземного маршрутного транспорта.

РАЗРАБОТКА ЯЗЫКОВ ВЗАИМОДЕЙСТВИЯ С СИСТЕМОЙ МНОГОКАМЕРНОЙ АВТОМАТИЗИРОВАННОЙ СЪЕМКИ

И.А. Герман

факультет Автоматики и вычислительной техники

Известно, что практически любая система строится по принципу преобразования некоторого количества определенных входных данных в выполнение своей функциональной деятельности. Для того, чтобы система могла преобразовать информацию, вид в котором она приходит, должен быть понятен системе.

В современном использовании можно выделить следующие наиболее распространенные способы передачи информации системе:

1. Ввод команд с клавиатуры и мышки
2. Голосовое управление
3. Управление жестами (с использованием датчиков распознавания)
4. touch управление

Понятно, что систему теоретически можно обучить распознавать любые команды, доступные для данных интерфейсов передачи. Однако создание языка команд интуитивно понятного пользователям конкретной системы есть одна из наиболее приоритетных задач, потому что удобство использования и составляет во многом оценку о системе.

Требования к языку команд:

1. Интуитивно понятен
2. Достаточность
3. Помехоустойчивость
4. Возможность максимально короткого оставления команды

Интуитивно понятный язык:

В режиссуре используются специфические выражения, фразы для передачи указаний к действию. Построение фраз должно максимально приближаться к стандартному построению предложений в русском языке

Достаточность языка:

Известный факт: в некоторых языках некоторых стран есть

слова, которые не встречаются у других народов. Однако это совсем не мешает другим народам безщербно коммуницировать. В рамках данной работы предстоит определить достаточность правил построения языка.

Помехоустойчивость:

Не редки ситуации, при которых теряется некоторое количество данных, например, если во время произнесения команды раздался громкий звук. Данную проблему можно решать, используя аналоговые средства удаления лишних шумов, однако правила построения языка также должны решать данную проблему.

Короткие команды:

Ситуация всегда может выйти из под контроля. Язык должен предусматривать набор специальных команд для экстренных ситуаций.

Распознание команд режиссера трансляции: Возьмем за основу несколько фактов:

Кейс – определенная ситуация, в которой происходит трансляция. Кейсы разделяются на «заданные технически» и «заданные режиссером»

Заданные технически – технически заданные кейсы система получает автоматически *при запуске* (расчет расстояния между камерами, расчет объема помещения) или *при событии* (выступающий вышел за трибуну: система рассчитывает рост выступающего, фокусное расстояние до него и т.д.)

Заданные режиссером – заданные режиссером кейс определяет характер съемки. Например, диалог «А» с «Б» (система меняет характер переключения и крупности планом таким образом, чтобы осуществить правильный монтаж диалога). Заданный режиссером кейс заранее описывается в сводной таблице и распознается системой при запросе.

Атрибут – параметр, описывающий кейс (например: форма зала, количество зрителей. (список атрибутов смотрите в сводной таблице))

Атрибуты разделяются на:

Необходимые атрибуты – без которых система не всегда сможет понять, что делать. Например, для кейса «Диалог», необходимыми атрибутами будут «А» и «Б»

Добавочные атрибуты – задаются режиссером. Например, для кейса «Диалог» атрибут «без «С»» будет добавочным

1. Элементарный запрос – минимальный технический запрос к системе («камера вправо на 5°»)

На элементарных запросах строятся атрибуты и в конечном итоге кейс. Такие запросы являются аварийным вариантом, когда управление системой переводится под «ручное управление»

Для таких запросов также предусмотрены голосовые команды

Формирование запроса к системе происходит либо с использованием атрибутов, либо с использованием заранее подготовленного кейса, либо с использованием «элементарного запроса»

Структура голосового запроса к системе:

Кейсовый запрос:

<кейс>Диалог<attributes>«А» с «Б» без «С»</attributes></кейс>

Атрибутный запрос:

<attributes> камера1 общий план, камера2 крупный план «А», камера3 крупный план «Б» </attributes>

Отображаемая информация:

Для возможности регулирования и правильного распознавания речевых запросов к системе требуется отображение следующих параметров:

- Начало приема, конец приема запроса