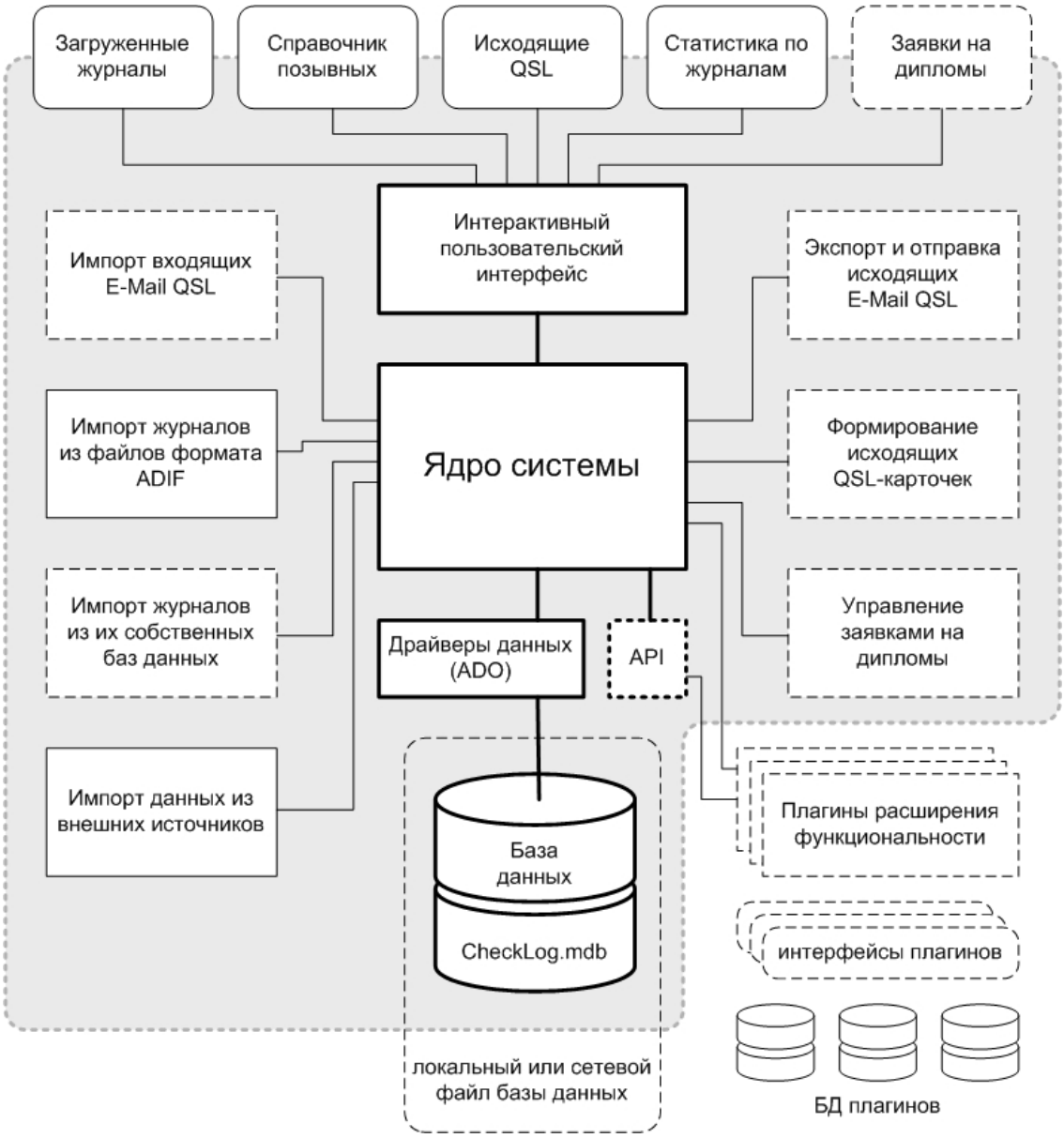
О назначении системы

Система анализа журналов радиолюбительской радиостанции CheckLog предназначена для углубленного анализа данных, накопленных в журналах станции, облегчения работы с массивами QSL-подтверждений и с заявками на радиолюбительские дипломы. Система разрабатывается мной, Дмитрием Речкиным (R2ADF) с использованием собственного времени и собственных ресурсов. Начиная разрабатывать ее как средство удовлетворения собственного любопытства, в беседах с другими радиолюбителями я обнаружил, что она может представлять интерес и для широкой радиолюбительской общественности. Поэтому пеед началом публичного тестирования системы я должен кое-что разъяснить…

Об архитектуре системы

Следующая схема на сегодняшний день является наиболее подробным отображением ее архитектуры. В сером поле показаны модули и блоки, реализацией которых занимаюсь лично я. То, что находится «за пределами» этого серого поля, может быть реализовано любым желающим в соответствии с его желаниями и потребностями. Необходимые средства и правила (документация для разработчика) будут опубликованы. Со временем… когда руки дойдут.



В варианте полного развертывания система анализатора журналов будет включать:

- ядро, обеспечивающее основную функциональность;
- интерактивный пользовательский интерфейс, реализованный через меню, многостраничное представление данных (включающее управляемые фильтры) в виде таблиц, и справочную подсистему;
- необходимые для поддержки работы с базой данных (ADO) драйверы;
- интерфейс для подключения функциональных расширений – плагинов (API).

Страницы пользовательского представления данных включают:

- отображение данных загруженных журналов, отсортированных по дате/времени QSO;
- отображение внутреннего справочника позывных (коллбука), отсортированного по позывным, с возможностью наложения пользовательских фильтров для облегчения поиска необходимой информации;
- отображение списка исходящих QSL (в список попадают QSO, не отмеченные в загружаемых журналах как связи с отправленными QSL), с возможностью наложения пользовательских фильтров;
- отображение суммарной статистики, формируемой по всем загруженным журналам, либо по выборке из них. С возможностью наложения на выборку пользовательских фильтров;
- отображение списка заявок на дипломы, содержащих формальные правила (условия выполнения) и таблицы подтвержденных связей, используемых для формирования соответствующих заявок [в настоящее время не реализовано].

Об использовании данных

Система ориентируется только на данные, импортируемые автоматизированными процедурами (пакетными либо однократными). Ручной ввод не предусматривается. Импорт данных осуществляется:

- из внешних источников (веб-ресурсы или файлы специализированных форматов), таких, как сервер Logbook of the World (LotW), сервер eQSL.cc, сервер поддержки UR5EQF.ru и подобных;
- из файлов универсального обменного формата ADIF (это путь загрузки данных, предварительно экспортированных из рабочих журналов);
- из собственных баз данных рабочих журналов [в настоящее время не реализовано];
- из внешних источников (аккаунтов на серверах электронной почты) либо массивов входящих писем, содержащих форматированное в соответствии с принятыми требованиями представление E-Mail QSL (карточек подтверждения, в виде строк ADIF-описаний и графических образов PNG, JPEG, GIF) [в настоящее время не реализовано].

Результаты анализа и обработки загруженных данных могут быть экспортированы в виде:

- исходящих писем, содержащих строки ADIF-описаний и автоматически генерируемые по заданным шаблонам графические образы QSL-карточек, E-Mail QSL [в настоящее время не реализовано];
- печатаемых QSL-карточек в традиционном (бумажном) представлении, включая формат стандартной карточки 140x90 мм, формат почтовой карточки (открытки) с заполненным почтовым адресом получателя и места для марки, надпечатка конверта для прямой

почтовой отправки QSL-карточки с заполненным адресом получателя, и т.п. [в настоящее время не реализовано];
- заявок на дипломы с прилагаемыми к ним выписками из аппаратного журнала станции [в настоящее время не реализовано].

Данные в универсальных форматах (например, в виде листов и книг Microsoft Excel) из системы не будут экспортироваться никогда. Хотя, конечно, желающие могут разработать расширение (плагин) специально для таких целей…

Все используемые ядром системы данные находятся в базе данных (формат ADO), единственный файл базы данных CheckLog.mdb может находиться как на локальном диске компьютера пользователя, так и на сетевом диске (внутри рабочей группы либо домена Windows). Допускается размещение системы и базы данных на съемном носителе.

О функциональных расширениях

Система предусматривает возможность подключения плагинов, дополнительно расширяющих базовую функциональность. Для доступа к функциям ядра предусматривается специализированный интерфейс (API), слабо зависящий от инструментальной системы разработки. Через предоставляемый API плагины получают возможность обращения к стандартными представлениям данных, выполняемым в ядре выборкам и фильтрам. Этот же API должен использоваться плагином для возврата в систему результатов обработки данных внутри плагина. Плагины стандартно вызываются через главное меню программы, при необходимости они могут иметь собственный интерактивный пользовательский интерфейс и собственные базы данных.

Об установке системы на компьютер

Для установки системы на компьютер пользователя требуется выполнить следующие шаги:

1). Создать в желаемом месте на диске компьютера подкаталог для размещения программы (например, D:\CheckLog)

2). Скопировать все файлы из дистрибутивного архива CheckLog.zip в созданный подкаталог.

3). Выполнить первый запуск, при этом ввести ключ активации и убедиться, что система выполнила автоматическое создание файла базы данных в подкаталоге DB.

Система не требует авторизации пользователя ни для начала работы, ни для подключения к базе данных. Необходимые параметры конфигурации записываются в файл CheckLog.ini, размещаемый в том же подкаталоге, что и основной исполняемый файл CheckLog.exe. В том же подкаталоге размещаются все необходимые служебные файлы (например, файл ключей активации). Файлы, как правило, имеют текстовый формат. База данных размещается в подкаталоге DB, создаваемом автоматически как вложенный подкаталог в каталоге размещения программы. Никакой записи информации в системный реестр не производится.

Для полнофункциональной работы с системой требуется ключ активации, жестко привязываемый к используемому позывному. Внимание: позывные считаются различными при любом несовпадении символов (например, R2ADF и R2ADF/A – это два различных позывных; в таком случае требуется два ключа активации). Пользователь может использовать для работы любое количество ключей активации. Количество поддерживаемых журналов не ограничено.

О том, что ждет пользователей с просроченным ключом активации

Система будет работать и в том случае, когда срок действия ключа активации истек. При этом в базе данных сохраняется вся накопленная информация, которая может без ограничений использоваться и обрабатываться ядром. Однако ввести новые данные при просроченном ключе не удастся.

О распространении системы

Сам дистрибутивный архив распространяется свободно без ограничений. За ключами активации любой лицензированный радиолюбитель (или сертифицированный наблюдатель) может обратиться непосредственно ко мне, личным запросом по электрической почте (в теме письма обязательно следует указать – CheckLog). Для сформированной группы бета-тестеров ключи выдаются мною же на любое количество вариантов личного позывного бесплатно, но с ограниченным разумными пределами сроком действия. Позывной и срок действия «зашиты» в ключ. Ключ кодирован. Способ кодирования достаточно формальный, чтобы быть простым в реализации и не требовать никаких лицензий на применение в составе выпускаемого продукта, в том числе и при целенаправленном либо случайном перемещении дистрибутивного архива за пределы нашей страны. В то же время способ кодирования достаточно «стойкий», чтобы на его понимание у неподготовленного человека ушло слишком большое время. Так что лучше не пытаться «подпольно сломать», а сразу связываться со мной.

О будущих планах (и о языке интерфейса)

Буду доделывать то, что объявлено в архитектуре. Готов сотрудничать (со временем, когда пройдем тестирование) с любым желающим попробовать себя в написании плагинов – и документацию предоставлю, и «каркас» плагина для использования в инструментальной системе разработки. Однако азам программирования учить не буду, равно как и помогать «заставить плагин работать». На каких условиях будут распространяться такие плагины – решать их разработчикам. Однако льготных ключей для разработчиков не будет, все и для всех, кроме сформированной группы бета-тестеров – на общих основаниях.

Интерфейс всегда (по крайней мере, в обозримом будущем) будет английским. Вмешаться в программу и «переправить» его на русский не удастся – на заложенные вовнутрь тексты предусмотрена программная реакция. «Отъедут» надписи – просто перестанет работать.

О целях тестирования

Самым распространенным заблуждением среди публики, имеющей возможность наблюдать за разработками программного обеспечения («софта»), с годами стало «Там нет ничего сложного, а программисты, как всегда, напутали».

Так вот, в этой формулировке неправильно все. Отчего так? Разберемся. Даже если программа выполняет ровно одну функцию – печатает на листочке бумаги текущее время и номер для какой-нибудь очереди, внутри нее скрыт гораздо более сложный мир. По крайней мере, она должна где-то брать значение точного времени и не слишком редко его обновлять. Она должна следить за заряженными в аппарат рулончиками той же бумажной ленты для печати – чтобы были, чтобы не перекосило, чтобы хватало краски для печати, чтобы проходила тем трактом, каким положено… и прочая, и прочая, и прочая. И окажется, что для успешной работы всего одной функции необходимо много, как это принято говорить «на англо-русском языке», пререквизитов. Наличие программиста является важным, но лишь последним звеном в цепочке. А шишки традиционно сыплются не на нерадивых снабженцев, кладовщиков, бухгалтеров, механиков, водителей, экспедиторов, монтажников, ремонтников и прочих «скрытых от глаз» – но именно на программистов. Каковые программисты, как бы хороши не были, сами и в одиночку этих проблем не решат.

Кроме того, программист может элементарно не представлять всех организационных и технологических тонкостей процессов, для которых он пишет программу. Ну ладно, в данном конкретном случае я – все-таки радист. Любитель, но радист. И я задумал этот проект как удовлетворение собственного интереса. Однако есть другие люди, интересующиеся моей программой. А у этих людей может быть другая привычка к той же радиолюбительской деятельности, которую я не видел и не учел при разработке «софта». А хорошая программа, как зелено-серый доллар, должна быть интересна и полезна всем. Ну, или большинству. Из интересующихся, конечно – папуасам из Мумбы-Юмбы доллары не нужны, им лишь «культ Карго» подавай.

Значит, главной целью тестирования является проверка поддерживаемой программой логики и технологии – на соответствие типичным приемам и привычкам работы людей. Второй целью выступает нахождение тех ситуаций, в которых программа ведет себя «неподобающе», не отвечая запросам и потребностям все тех же заинтересованных людей.

Понятно, что один лишь разработчик этих целей не достигнет никогда.

О задачах тестирования

Как говорится, «Цели ясны – за работу, товарищи!». Определимся, какие задачи мы решаем во время тестирования.

Задача первая, главная и основная: пройти всю логику программы, поработать во всех режимах, использовать все функции во всевозможных условиях и в самых разнообразных окружениях. Сюда входят не только особенности применяемого компьютерного «железа», но и многое другое: операционная система, настройка локализации (национальных и региональных особенностей применения), скорость подключения к Сети (как локальной, так и Интернет)… Всего не перечислишь, но это чрезвычайно важно – чтобы в дальнейшем можно было одной фразой сказать, что «оно работает на всем, что доступно». И чтобы этой фразе поверили.

Задача вторая, но от этого не менее важная: проходя логику, отрабатывать абсолютно все возможные варианты последовательности действий, абсолютно все «маршруты» достижения результата. То есть не «прогнать программу до конца», а воспользоваться всеми заявленными возможностями и убедиться – выполняет программа заявленные функции, или не выполняет. В конечном счете, чтобы заявить, что «оно делает все, что должно».

Третья задача, по степени важности и ответственности не отстающая от первых двух: строго следуя отработке всех функций и всевозможных вариантов использования, аккуратно фиксировать все ситуации, маршруты, наборы входных данных, сообщения программы и прочее – для того, чтобы с уверенностью можно было утверждать, что «оно не делает ничего такого, чего не должно».

Вот такие три задачи. Или «триединая задача тестирования». Коммунизма мы тут не построим, но к поставленным целям – приблизимся. А больше нам ничего и не надо.

Об инструментах и методах

Теперь, когда мы определились и с целями, и с задачами, надо договориться о методах и средствах для их достижения/решения.

Тестируя программу, никогда не следует поддаваться соблазну взять да и потыкать кнопки, погонять все и сразу, только для того, чтобы убедиться – эта программа «всосала» мои данные, нарисовала мне пару-тройку таблиц, чего-то там посчитала и при этом ни разу не «свалилась» и не «зависла». Тем более, что такое жаргонное использование привычных слов по отношению к миру программ – просто переворачивание привычного «с ног на голову». В обычном мире слово «умерла» означает – все, кончилась. Прекратила всякую жизнедеятельность. А для программы – она живее всех живых, она забрала на себя все ресурсы процессора, она «шуршит» диском непрерывно, не оставляя другим программам шанса хоть чего-то урвать себе от предоставляемых компьютером и операционкой ресурсов. Привычное слово «встало» большинство воспринимает как приведение в полную готовность того, чем сейчас будут втыкать, ковырять, долбить или наворачивать. А программа, которая «встала» – просто оказалась не готова к дальнейшей работе. То есть скорее «легла»… Хотя и здесь есть варианты…

К чему это я? А все к тому же, к инструментам и методам.

Основным инструментом тестировщика является его родной язык. Формулируя на нем (и только на нем!) описания ситуаций, маршруты действий, корректность вычислений и полноту отработки функций, он не должен скатываться в жаргон, как бы этого ни хотелось. Описания следует делать в законченной (хотел сказать – литературной) форме. По возможности – не допускающей двоякого понимания и побочного истолкования.

Второй инструмент – это средство фиксации фактов. Точно записывать ход выполнения, сообщения программы, сохранять настройки, «фотографировать» формы по результатам выполнения шагов. Не путать буквы в описании команд, меню и кнопок. Точность – вежливость королей. Попробуйте, побудьте королями хоть недолго. Вам понравится.

Методов у тестировщика всего два. Поэтому освоить их и успешно применять сможет любой, кто проникся целями и задачами, про которые шел рассказ чуть раньше.

Метод первый: работа с документацией. Нужно скрупулезно и настойчиво проверять, все ли важное для работы описано в документации. Нет ли «пропущенных» мест, отсутствие которых

препятствует пониманию логики и условий работы программы. Нет ли логических ошибок, опечаток или просто неправильных описаний, из-за которых человек может совершать неверные шаги при работе (или вовсе не совершать необходимых действий).

Метод второй: работа с программой. Не надо слепо доверять программе, которая «что-то там делает». Надо проверять – в меню написано так-то, в документации это определено так-то. А в результате? Что я вижу на экране? Что произошло с моими данными? Есть ли невязки между тем, что должно быть, и тем, что на самом деле произошло? Попробовать «загнать программу в угол» – не самое главное, но это тоже необходимо. Если удастся выявить и зафиксировать, а потом еще и воспроизвести ошибочные ситуации, в которые программа позволяет себя загнать действиями пользователя – это удача. Не найденный в густой воде речки Яузы золотой самородок, но все же. Каждая такая выявленная во время тестирования ситуация позволит вовремя внести исправления в код программы, что сделает ее более надежной. Значит, другие будут меньше ожидать «подвохов» от программы в сложных ситуациях. То есть – будут меньше психовать. Заодно не так часто станут искать виноватых среди программистов. А душевное здоровье в наше время дорогого стоит, да.

О «черном ящике»

Тестировщики получают не исходные тексты программы, а уже готовый, откомпилированный код в виде исполняемого файла. Они не могут «заглянуть» внутрь, и разобраться – что там наворочено в логике работы, какие условия проверяются, какие ситуации игнорируются. И я считал, считаю и буду считать – это правильно. Так и должно быть. Потому как прямой доступ к исходному коду вызывает соблазн при обнаружении первой же ошибки тут же «полезть внутрь» и сразу чего-то там наскоро переписать. Переписывая одно, неизбежно затронешь другое. Исправление одной ошибки наспех и без необходимого анализа взаимосвязей в системе приведет лишь к появлению новых ошибок. Я видел много программистов (и сам грешен, не скрою), которые из-за «мании последней поправки» ломали уже почти до конца доведенную логику работы сложных систем. В результате месяцы и годы работы просто «шли в помойку». Поэтому, как мантру, тестировщику надо заучивать: «Я не знаю, ошибка это или нет. Я лишь могу свериться с описанием, зафиксировать разницу между написанным и произошедшим, и донести ее до разработчика». А потом, после внесения исправлений и выпуска новой версии – опять проверить ее на «знакомых местах», чтобы убедиться в отсутствии выявленных ранее несоответствий.

О наблюдательности

Это уже совсем просто. Глядя на экран или на распечатку, надо стараться держать глаз «не замыленным» от неоднократного разглядывания все того же во время предыдущих запусков программы. Не надо уверять себя в том, что оно должно происходить именно так, как происходит. Живой пример: при отладке именно этой программы трое человек «прозяпали» очевидный эффект пропадания русских букв в именах и географических названиях. И я в том числе. Спасибо, в нашей команде появился четвертый участник, который ткнул меня носом в этот очевидный «косяк». Теперь это поправлено, но мы еще долго будем вспоминать, как не замечали самого очевидного, привыкнув уже к пользовательскому интерфейсу в его внешнем изображении. Привыкнув до того, что перестали читать показываемое на экране. Такие дела.

Об аккуратности

Аккуратность, аккуратность и еще раз аккуратность. Никогда не начинать тестирование, не положив перед собой листок бумаги для заметок. Почему листок, а не открытый документ в компьютерном редакторе? Да потому, что листок с карандашом не могут внезапно «зависнуть». То есть вы всегда сможете записать необходимое, невзирая на жизнедеятельность компьютера и жизнеспособность запущенных программ. А записывать надо, причем много. Возьмите за правило: сначала записывать шаги, которые вы хотите выполнить, и лишь потом их выполнять. А выполняя – отмечать, какие прошли нормально, а какие – нет. Что такое в этом контексте «нет» – ну, мы говорили об этом раньше, посмотрите «Об инструментах и методах»…

Кому это нужно? Кому это выгодно?

Это и нужно, и выгодно – вам, прежде всего. Потому как разработчик уйдет, а программа останется. И, насколько успешно проведено тестирование, определит – будет ли программа многолетним надежным помощником в вашей жизни, или отойдет в небытие, оставив вас, как и раньше, наедине со своими проблемами. А во время проведения тестирования у вас, и именно у вас есть уникальная возможность повлиять на исходную постановку задачи, тесно сотрудничая с разработчиком, в конце концов, вместе создать инструмент, который будет удобным, полезным, надежным… да и просто красив, в конце-то концов. Ведь иметь под рукой красивую вещь – само по себе удовольствие. Ну, и возникающее самоощущение того, что довелось «постоять у истоков»…

Здесь мы говорим о нашем хобби. Но разве мы согласимся променять то душевное успокоение и чувство насыщенной интересной жизни, получаемое от него, на нервотрепку и переживания всего лишь из-за какой-то программы, вовремя не «доведенной до ума»?

Тестирование идет уже больше четырех недель. Мы (то есть разработчик и команда тестировщиков) проделали невероятную по объему работу, пройдя вместе путь, на преодоление которого в других условиях понадобилось бы в разы больше времени. Месяцы, а может – и годы. «Однако за время пути» накопилось много вопросов, и прежде всего – вопросов, определяющих самые основы нашей деятельности. Возможно, я расскажу не обо всех. Меня могут прервать, заявив: «Настоящий гражданин должен быть "враг всех так называемых вопросов"» – увы, Козьма Прутков успел заявить такое задолго до вас, уважаемые. Так что – вы нам не мешайте, а мы – приступим. Тем более, что к нашей команде присоединяются новые люди.

С чего начать?

И который раз повторяю: с документации! *«В начале было Слово»* [Ин.1,1] Все, что программа должна была делать – было сначала записано, пусть и не в красиво оформленном виде. Однако сначала – замысел, а уже потом – реализация. Иначе продукт, способный заинтересовать многих пользователей, просто не может возникнуть. Иначе будут только попытки «что-то такое сотворить, от чего другие заметят пользу для себя и скажут "спасибо".

Документация, документация, и только документация! Во всяком случае, в начале тестирования. Прочитать, понять. То, что непонятно – переспросить. Не удовлетвориться частным ответом, а потребовать, чтобы эта информация (и вопрос, и ответ, в любой форме) была обязательно включена в документацию. Если нашелся один, кому непонятно – будут и другие. Значит, записанные слова должны быть понятны всем (клинические случаи исключаем).

Что и как делать?

Первое. Читать документацию и составлять план тестирования. План этот должен быть простым, чтобы самому в нем не путаться. В то же время он должен быть подробным, чтобы не пропустить ничего важного и существенного, «заглянуть во все уголки», руководствуясь им. Документация пишется для пользователя. Тестировщик – тоже пользователь, но только во вторую очередь. А в первую очередь он – приемщик того, что «наплодил разработчик». Собственные фантазии и хотелки могут завести разработчика так далеко от насущных потребностей пользователя, что просто ох и ах! *«Блуждает человек, пока в нем есть стремленья»* [Гёте, "Фауст"].

Второе. Строго следовать плану тестирования. Не отклоняться на частности, выполнять пункт за пунктом, как это определено. Нашли что-то, не отраженное планом? захотели попробовать? – немедленно переходите к пункту первому. Есть в документации? запланировано для проверки? – если нет и нет, отложите. Запишите как свои пожелания к продукту, но не отвлекайтесь от главного. Иначе – блуждания, сопровождаемые не только сомнениями, но и просто потерей времени. *Nel mezzo del cammin di nostra vita // mi ritrovai per una selva oscura // ché la diritta via era smarrita*.—ой, не удержался! Но красиво-то как! По-русски это же*: «На полпути земного бытия // Вступил я в лес угрюмый и унылый, // И затерялась в нем тропа моя»* [Данте, "Божественная Комедия"].

Третье. Последовательно фиксировать результаты. Записывать, что происходит. Не надо переписывать все, что показывается на экране. Достаточно скопировать содержимое окна сообщений, или «сфотографировать» рабочее окно стандартными средствами операционной системы. Не надо понимать слово «снимок» буквально и фотографировать фотоаппаратом или телефоном – такие снимки просто ужасающе велики и способны вызвать только досаду. А

информации в них столько же, сколько в трех строках текста. И – непременная аккуратность, выражающаяся в фиксации всего, достойного внимания. *«Повторяю, я вёл мой дневник аккуратно»* [Дефо, ”Жизнь и необычайные приключения Робинзона Крузо…”]

Хочу ли я? Могу ли я?

Резонные вопросы. Желание первому пройти путь, который до этого полностью не проходил никто – вот единственная мотивация, которую я ожидаю увидеть в каждом, входящем в команду тестировщиков. Если желания нет – ну, что поделаешь? Это не работа по найму, и не принуждение. Никто никому ничем не обязан – вот главное. Пропало желание – уходите. Жаль, конечно, будет расставаться… но уж так устроен человек, не будет он хорошо работать «из-под палки».

А вот второй вопрос – очень опасный. И по-настоящему бестактный по отношению к тому человеку, который его задает. Он ведет к самоуничижению, к неверию в свои силы, знания, опыт и способности. Задайте себе другой вопрос: «А что мне мешает?» Сразу станет понятнее – не хватает каких-то знаний, надо понабраться навыков, попробовать себя. Так, собственно, именно за этим люди и тянутся к приключениям. Смотрите на все происходящее, как на очередное приключение в вашей жизни – и не опускайте рук! По сути, вы ничем не рискуете. Зато достигнув результата, будете еще долго вспоминать, как это было. Ручаюсь, у большинства из нашей команды подобного в жизни еще не было – ну неужели же не хочется попробовать?

Есть такое – ”идеи чучхэ”. Там много странного и смешного, но вот главный тезис я оспаривать не возьмусь: *«Человек хозяин всего, и он решает всё!»* Неплохо, да? Запомните, пригодится…

Как правильно?

Правильно – это когда вам нравится. Если приходится не решать насущную задачу, а постоянно «воевать» с программой – это неправильно. Надо спрашивать себя – хорошо ли, плохо ли. Нравится мне, как она (программа) работает? – хорошо. Не нравится? – плохо.

Сколько людей – столько и мнений. У особо отмеченных народов на два человека вообще приходится три отдельных мнения, и ничего! Зато в понимании «красиво» – «не красиво», как ни странно, сходится большинство людей. Если интерфейс программы красив – скорее всего, он для вас окажется удобен. Красив – но не из-за всяких «моргалок», «пищалок» и «давилок»… Красив, потому что функционален. Лаконичен. Интуитивно понятен.

Поэтому не стесняйтесь высказывать претензии и пожелания к интерфейсу. Не хватает каких-то элементов управления – сообщите. Функциональность элементов управления хочется расширить – дайте знать. Отображение данных не включает чего-то важного? – заявите об этом. Был только один, кто самостоятельно мог определить качество своих творений. *«И стало так. И увидел Бог все, что Он создал, и вот, хорошо весьма»* [Быт.1,30-31]. Мы – не боги. Мы просто обжигаем свои горшки. И хотим, чтобы эти горшки служили долго, были удобными и красивыми. И не только для нас, любимых. Как-то так…

А что, если…?

А попробуйте! Для того она и программа, чтобы попробовать получить от неё всё, на что она способна. А вдруг окажется, что можно ещё получить и то, чего в документации не указано? А

вдруг это окажется тем «бантиком», от которого решение моей задачи засияет особенным светом, даст мне новое понимание и новые возможности?

Дерзайте! Загоните программу «в угол» – тоже польза, будет понятно, как строить в дальнейшем «защиту от дурака». Только не забывайте всё фиксировать, и перед отправкой разработчику – обязательно воспроизведите ситуацию еще и еще раз. Будет уверенность, что это не случайно – будем исправлять. Неоценимую помощь окажет точное описание последовательности «ходов», от запуска программы до ее «краха» или «зависания». *«И предал я сердце мое тому, чтобы познать мудрость и познать безумие и глупость»* [Еккл.1,17]

Когда, наконец, всё это закончится?

Когда-нибудь вообще всё закончится. Последняя масса, наконец, будет втянута в последнюю гигантскую «чёрную дыру». Примерно тогда же прекратится всякое движение, и наступит «тепловая смерть Вселенной». Несколько раньше погаснет последняя звезда. Еще чуть раньше наше Солнце раздуется до таких размеров, что поглотит Землю. Перед этим кончится вода, магнитное поле, электричество и нефть. Задолго до этого будет «конь бледный и всадник на нем бледный», и что-то такое [Откр.16,16] близ Хар Мегиддо… (32°35’06”N 35°11’03”E, **KM72on**)

А пока всего этого не произошло, мы стараемся сделать свою работу так, чтобы другие, наконец, могли в полной мере вкусить плоды её. И всласть попользоваться тем, что мы своими трудами создаём.

Надеюсь, к марту (или чуть раньше) сделаем…

А что потом?

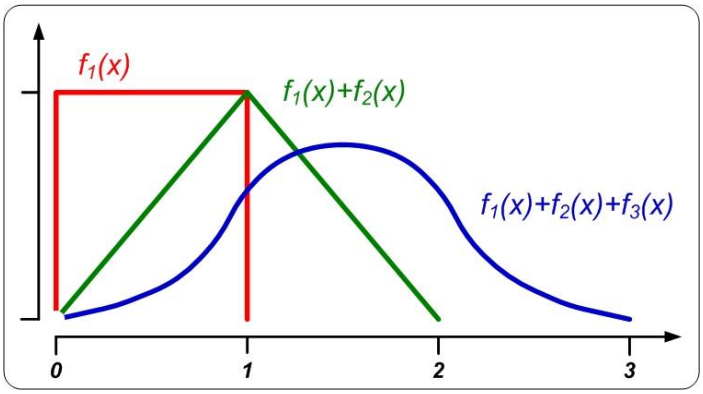
А потом, когда мы выпустим релиз анализатора, как ядро будущей системы – работа продолжится. Будет программа печати QSL-карточек, существенно помогающая в экономии бланков и сокращении расходов на рассылку – будем печатать для тех, кто в состоянии их получить, хотя бы. Будет система обмена E-Mail QSL со встроенной подписью и защитной маркировкой. Будет анализ выполнения дипломов, автоматизированное формирование заявок на них… А не хватит этого – ещё что-нибудь придумаем. Наши же благодарные пользователи нам в этом и помогут.

«Вы полагаете, всё это будет носиться? // Я полагаю, что всё это следует шить!» [Левитанский]

Любительские радиосвязи как проявление случайного процесса

Что говорит в пользу случайности в подтверждении состоявшейся радиосвязи? Радиолюбителей в мире достаточно много. Никто из них не в состоянии заранее управлять условиями, при которых излучаемый сигнал достигнет только тех корреспондентов, с которыми связь запланирована заранее – всегда существуют операторы, работающие «на поиск», и операторы, работающие на «общий вызов». На общем фоне таких связей теряются связи между двумя корреспондентами, договаривающимися предварительно о времени, частоте и используемой модуляции.

Используемые частотные планы разрешают проведение связей выбранной модуляцией в достаточно широкой полосе частот любительского диапазона. Поэтому распределение излучающих станций, вообще говоря, в пределах участка диапазона более всего соответствует равномерному распределению (функция плотности распределения $f_1(x)$ на графике):

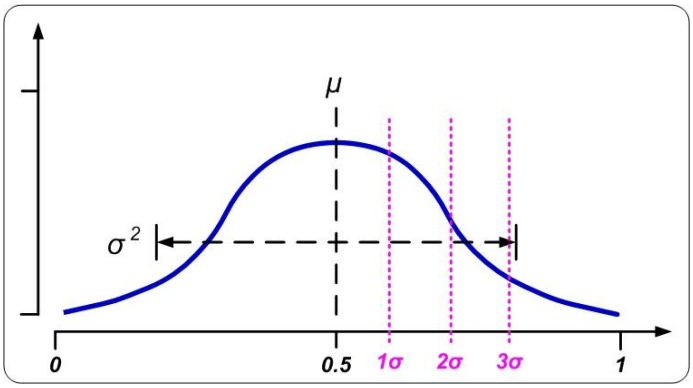


На практике обычно приходится иметь дело не с одной функцией распределения, а с несколькими независимыми. Так плотность суммы трех независимый функций $f_1(x)+f_2(x)+f_3(x)$ (соответствует выбору частоты, модуляции и времени радиосвязи) имеет выраженный «колоколообразный» характер, приближающий ее (сумму) к нормальному распределению (или распределению Гаусса).

Подтверждение радиосвязей как функция вероятности

Конечно, в основе отправки или неотправки QSL-карточки оператором по фактически состоявшейся связи лежит личное его желание (или нежелание). Однако для множества операторов, объединяемых теми или иными условиями выборки, действуют уже не столько личные желания, а проявления закона больших чисел. Если рассматривать подтверждения связей как результат случайного выбора действия по отправке карточки случайно выбранным оператором, то более всего характер распределения будет соответствовать уже упомянутому нормальному распределению. В качестве пределов изменения аргумента функции случайной переменной рассматриваем вероятность подтверждения связи – $P=0$ в случаях, когда связь не подтверждается ни при каких условиях, и $P=1$ в случаях гарантированного подтверждения связи.

Распределение вероятностей же имеет выраженный максимум в районе $P=0.5$, и заметный колоколообразный характер. Никто не волен «приказать» выполнять те или иные действия оператору любительской радиостанции «за пределами эфира» – так мы понимаем свободу выбора каждого оператора. Для такого распределения можно определить границы, в пределах которых вероятностные оценки будут достаточно надежными, чтобы использовать их в прогнозировании поведения других операторов, находящихся в похожих условиях.



Из «метрик» нормального распределения важны математическое ожидание ***μ***, и дисперсия ***σ²***. Зная всего эти две величины, можно определить функцию плотности вероятности исчерпывающим образом:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)^2}$$

Все узнали основание натуральных логарифмов и «число Пи»? Зная дисперсию ***σ²*** нормального распределения, можно легко определить среднеквадратичное отклонение ***σ*** (читается как «сигма»), имеющее смысл ошибки измерения случайной величины ***x***.

Интересным свойством нормального распределения является то, что в пределах ***–3σ…+3σ*** заключено не менее 95% всех случайных значений ***x***. Иными словами, выбирая эти границы, мы точно не ошибемся в оценке интересующего нас значения математического ожидания ***μ***.

Что же это за величина математического ожидания, если мы говорим о показателях подтверждаемости связей? Да всё очень просто – это вероятность того, что операторы данной выборки подтвердят нам проведенную связь. Вот так, ни больше, ни меньше.

Оценка подтверждаемости связей, от «хорошей» к «плохой»

Самыми «хорошими», разумеется, будем считать оценки, для которых показатель подтверждаемости ***X*** для выборки превысит среднюю подтверждаемость по всему оцениваемому множеству связей более, чем на «одну сигму». Соответственно, оценки показателя подтверждаемости, отклоняющиеся «в минус» более, чем на «одну сигму» – «плохие». А в промежутке «раскрасим» оценки от «умеренно плохих» до «умеренно хороших», руководствуясь все тем же отклонением выборочного среднего от среднего по всему массиву на те же кратные «сигме» ступени.

Ну, посчитали… и что с этим делать?

Ну… можно ничего не делать, так жить проще.

Однако, если есть желание провести как можно больше связей, нацеленных «на результат» – то есть на получение максимально возможного количества подтверждений при минимальных (ну, или оптимальных) затратах времени, денег и нервов… Тогда, конечно, надо смотреть на показатели подтверждаемости не после отправки собственных подтверждений, а до этого. Или вообще, до проведения связи. Услышав позывной, достаточно просто проверить, с какой вероятностью операторы, относящиеся к данной территории, подтверждают связи. А дальше

решать самому – если территория сверх-редкая, такой прямо «настоящий DX!», то не грех и провести связь, и послать собственное подтверждение, в расчете на взаимность далёкого оператора.

А вот если вызов шлёт житель той территории, где уровень воспитания не позволяет понять головой, что подтверждение проведенных связей – главная вежливость и добродетель радиолюбителя… проще промолчать, не отвечать на этот вызов, наверное… Или, по крайней мере, перед отправкой собственных карточек-подтверждений лишний раз проверить – будет ли на них ожидаемый отклик.

А где смотреть, кто «хороший», а кто – «плохой»?

Во-первых, кто хороший – а кто плохой, решает не статистика. Это уж вам самим решать, заслуживает ли представитель территории, отличающейся заведомой необязательностью и невоспитанностью, вашего внимания и усилий по проведению связи. Во-вторых, вы оперируете только собственным опытом, накопленным в ваших собственных журналах. Никто не навязывает вам своих (возможно, субъективных) оценок.

А программное обеспечение может только вовремя дать полезную информацию. При условии правильной предварительной подготовки данных. Но это уже – за пределами «добра и зла», и вне тематики этой лекции.