Дипломная работа

по специальности
«Программное обеспечение вычислительной техники и автоматизированных систем»

на тему **«Создание библиотеки функций унификации процессов обработки входных параметров и систематизации выходных данных в средствах тестирования и диагностики программных средств и оборудования»**

Дипломант: студент Гусев М.С.
Руководитель: доцент, к. т. н. Новиков П.В.

Москва 2012 г.

Постановка задачи. Предпосылки к разработке библиотеки функций

Целью дипломной работы является разработка библиотеки, решающей задачу автоматизации запуска, сбора информации и интерпретации результатов тестирования

Предпосылки разработки

Необходимый функционал	Недостатки существующих аналогов
• Унификация получения параметров • Интеграция в систему тестирования более высокого уровня • Потоковая безопасность • Малый размер и высокая переносимость	• Интеграция всех тестов в единый исполняемый модуль • Отсутствие средств унификации вывода табличной информации • Осуществление ввода данных через глобальные переменные

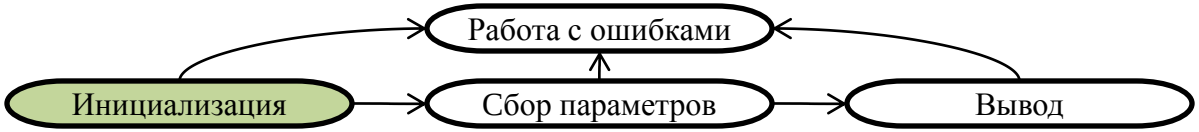
Каскадная модель проектирования библиотеки функций

Другие модели ЖЦ

- V-образная модель
- Модель RAD
- Спиральная модель
- Инкрементная модель

Причины по которым была выбрана именно эта модель:

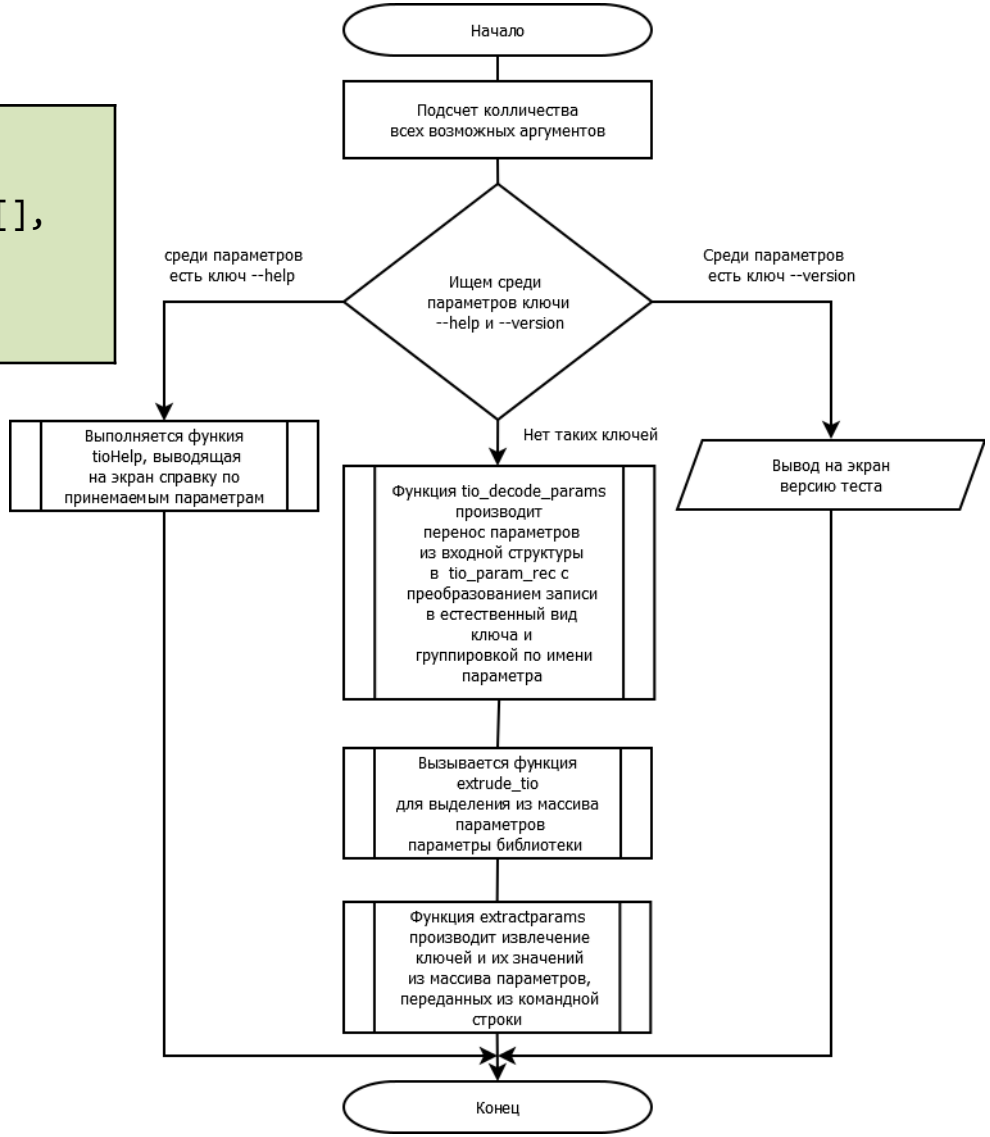
- Требования к работе ясны и понятны
- Сроки выполнения работы жестко ограничены



Функция инициализации библиотеки

```
int tioInit  (  const char* version,
                const char* help,
                const tio_param _param[],
                int argc,
                char *argv[]
             )
```

```
typedef struct _tio_param
{
    char *key;
    char *name;
    char* description;
} tio_param;
```



Функции получения входных параметров

- ☑ `int tioGetS(const char* name, char* buff, const size_t buff_len);`
- ☑ `int tioGetDefS(const char* name, const char* default, char* buff, const size_t buff_len);`
- ☑ `long tioGetL(const char* name);`
- ☑ `long tioGetDefL(const char* name, const long default);`
- ☑ `unsigned char tioGetC(const char* name);`
- ☑ `unsigned char tioGetDefC(const char* name, const unsigned char default);`
- ☑ `double tioGetD(const char* name);`
- ☑ `double tioGetDefD(const char* name, const double default);`

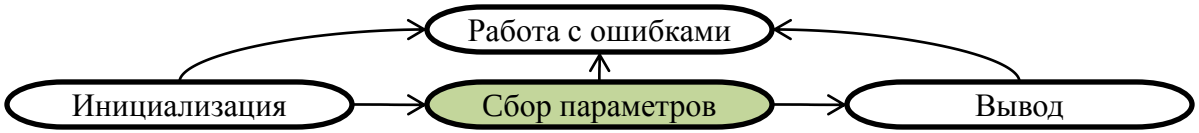
Суффикс Def говорит о том, что случае если параметр не будет получен, возвращается значение по умолчанию.

Последняя буква – суффикс, обозначающий тип возвращаемого значения.

- L – long
- D – double
- C – char
- S – char* (строка)

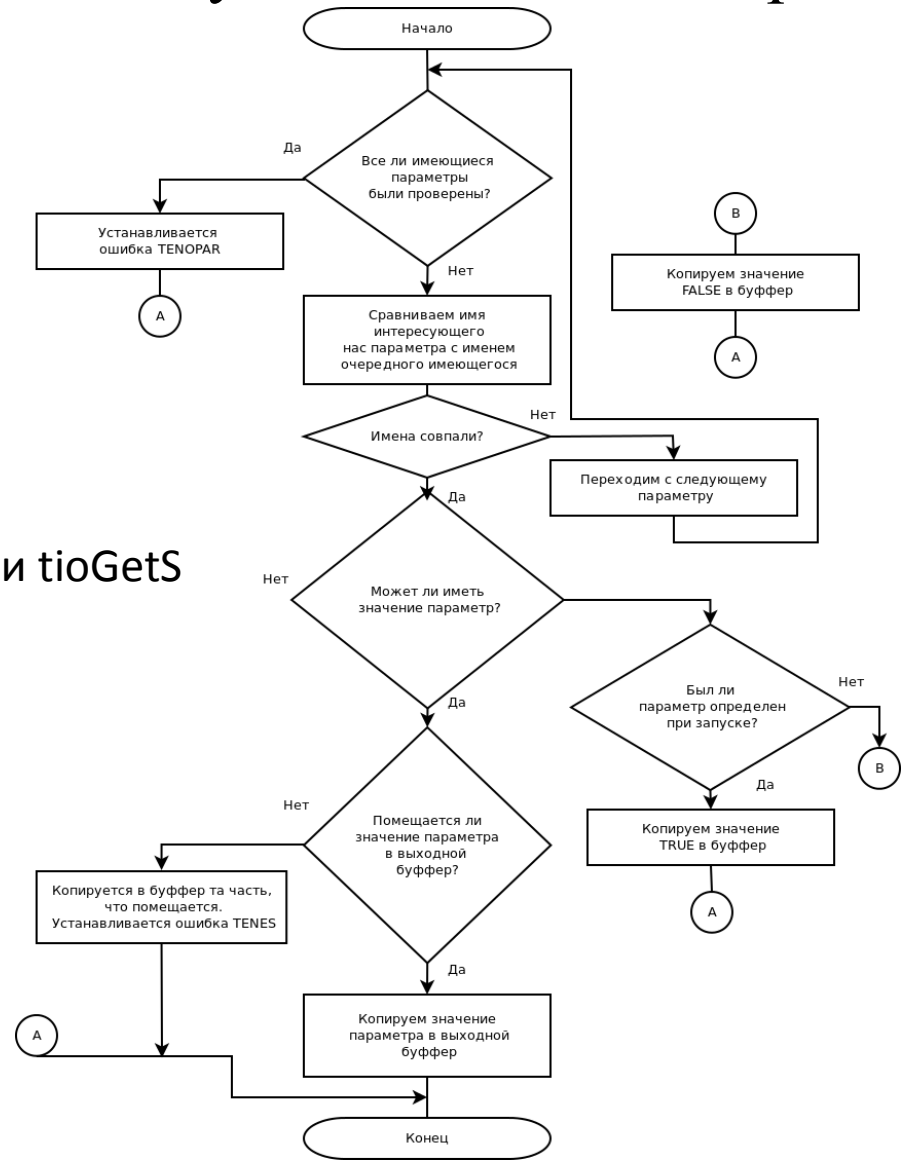
Коды ошибок

TENOPAR	Параметр не зарегистрирован при инициализации библиотеки
TEINCTYPE	Параметр не может быть приведен к запрошенному типу
TENOTSET	Параметр не передан при вызове приложения.
TENES	Размер буфера недостаточно велик для помещения параметра
TEFAILL	Отказ по непонятным причинам



Функции получения входных параметров

Блок-схема функции tioGetS



Функции работы с ошибками

int tioDie (int status, const char * msg)

Аварийное завершает работу приложения, выводит в поток ошибок сообщение **msg**. Параметр **status** не может принимать значение 0.

Коды параметра **status**

1 - тест провален
2 - не выполнены условия запуска теста
3 - внутренняя ошибка библиотеки

int tioGetError(void) Возвращает код ошибки для последней вызванной функции библиотеки.

Общие сообщения об ошибках

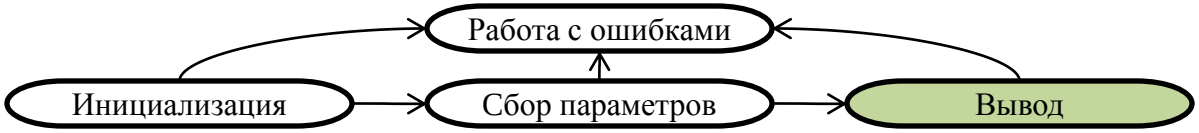
TESUC	успешное выполнение
TEKLEN	длинный ключ
TEFAIL	отказ приложения в системной части

Сообщения об ошибках при выводе

TENOFREEID	нет свободного идентификатора для структуры
TEINVAL	неправильные параметры функции
TEINTMC	непредвиденное состояние внутренних переменных библиотеки

Сообщения об ошибках при получении параметров

TENOPAR	параметр не передан в приложение при инициализации
TEINCTYPE	невозможно привести запрошенный параметр к тому типу данных в котором его запросили
TENOTSET	запрошенный параметр не установлен
TENES	переданный буфер не достаточного размера



Функции строчного и табличного вывода

Строчный вывод

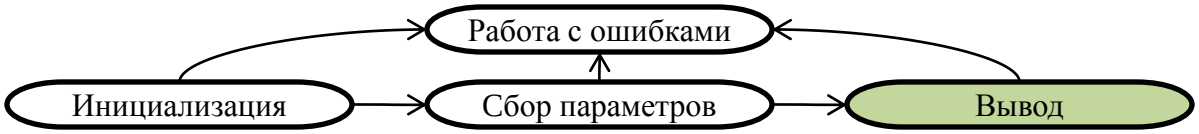
```
int tioPrint(const char * message)
int tioPrintF(const char* template, ... )
int tioWarning( const char * message)
int tioWarningF(const char* template, ... )
int tioError( const char * message)
int tioErrorF(const char* template, ... )

int tioDebug( const char * message)
int tioDebugF(const char* template, ... )
```

Табличный вывод

```
void* tioTableBegin ( const char* format, … )
void* tioTableRecord ( void *td, … )
int   tioTableEnd( void *td )
```

Последовательность символов для форматирования строки

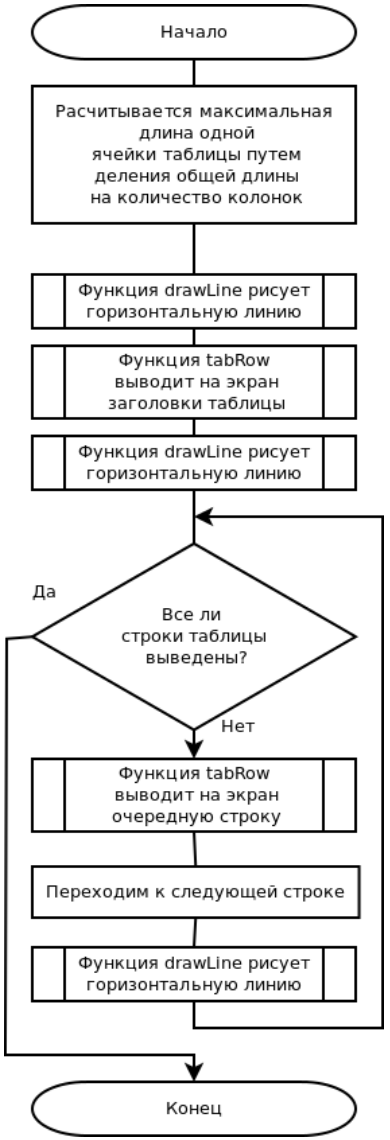
%c	Символ (char)
%d | i	Целое число в десятичной форме (long)
%e	Число с мантиссой для чисел с плавающей запятой (double)
%f	Число с плавающей точкой (double)
%o	Целое число в восьмеричном представлении (long)
%s	Строка завещающаяся нулем (char*)
% x	Беззнаковое шеснадцатиричное представления (long)
%X	Беззнаковое шестнадцатеричное представления с буквами в верхнем регистре (long)

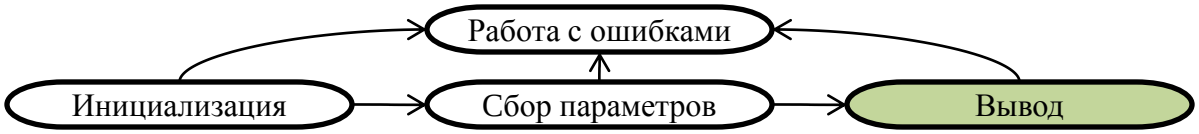


Функции строчного и табличного вывода

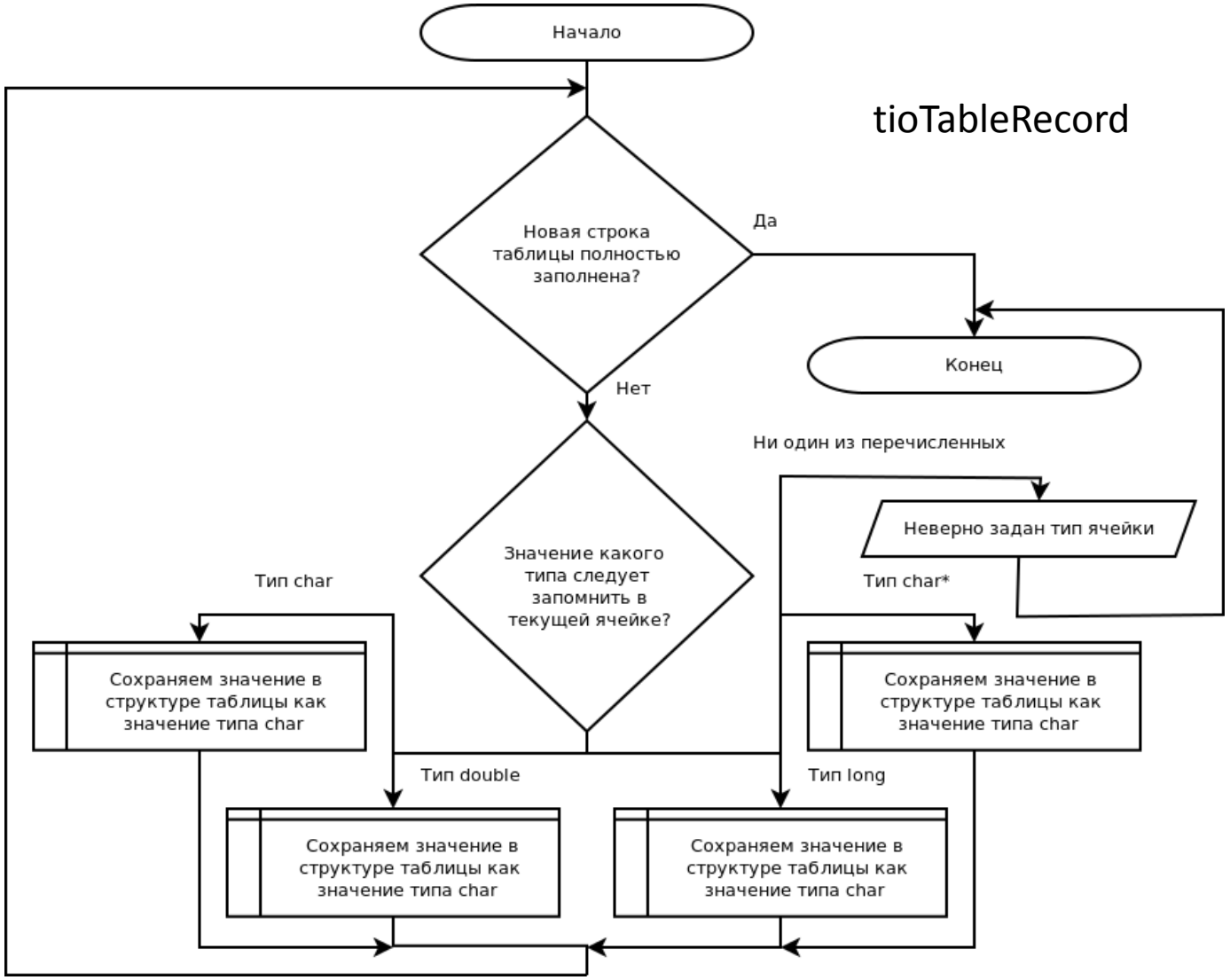
tioTableBegin

tioTableEnd





Функции строчного и табличного вывода



Демонстрация среды исполнения подпрограмм библиотеки

```
(TS): Run test: bin/test_help
Использование: Test [КЛЮЧ]... [ФАЙЛ]...
Проверка описания программы.
 -k, -d, --list, --ls, --lst    List information about the FILEs (the current dire
                                ctory by default).  Sort entries alphabetically if
                                 none of -cftuvSUX nor --sort. Mandatory arguments
                                 to long options are mandatory for short options t
                                oo.
 --ip, --adress <ПАРАМЕТР>      show / manipulate routing, devices, policy routing
                                 and tunnels
 -s, -S, --sort <ПАРАМЕТР>      sort lines of text files
 --file, --fl <ПАРАМЕТР>        determine file type
 -t <ПАРАМЕТР>                  table view
(TS): Test  bin/test_help [PASS]
(TS): Run test: bin/test_table
Cap string
Call "tioTableBegin".

Call "tioTableRecord".

Call "tioTableEnd".
+----------------------------+----------------------------+----------------------------+----------------------------+
|char                        |st&ring                     |double                      |string                      |
+----------------------------+----------------------------+----------------------------+----------------------------+
|r                           |An advantage of COM+        |23.70                       |Animated by Ryan Woodward   |
+----------------------------+----------------------------+----------------------------+----------------------------+
|e                           |An advantage of COM+        |43.90                       |The essence of COM is a langu|
|                            |                            |                            |age-neutral way of implementi|
|                            |                            |                            |ng objects that can be used i|
|                            |                            |                            |n environments              |
+----------------------------+----------------------------+----------------------------+----------------------------+
(TS): Test  bin/test_table [PASS]
(TS): Run test: bin/test_tioInit
Test start
[RUN]: Запуск self
[PASS]: self : Тест пройден успешно
(TS): Test  bin/test_tioInit [PASS]
```

Фрагмент вывода автоматического модульного тестирования функций библиотеки

Демонстрация среды исполнения подпрограмм библиотеки

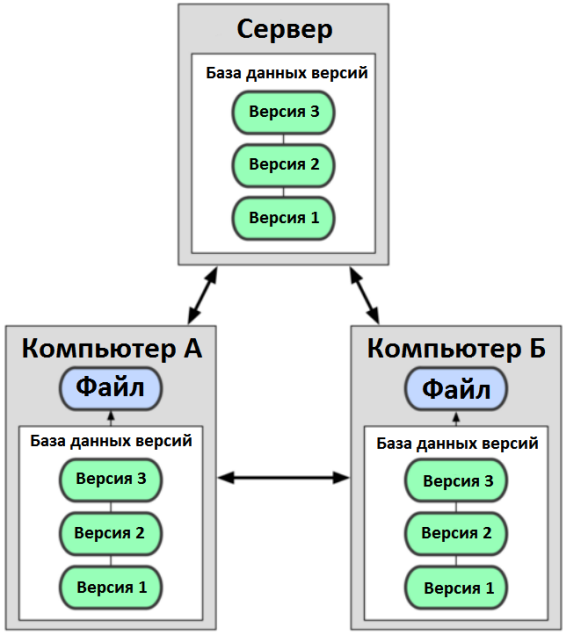
```
nwcfang@nwcfang-Z68AP-D3:~/development/rti/rs232test$ sudo ./_test_COM -L /dev/ttyS0
[sudo] password for nwcfang:
[RUN]: Запуск ./_test_COM
(DD)[client.c@228]: Starting server wait
(DD)[client.c@53]: Current 1354726090: stat at 1354726090: elapsed: 0
(DD)[server.c@162]: Starting server process
(DD)[client.c@77]{readBuffer}->{Found recive buffer: 1FFF9AA9}
(DD)[client.c@236]: Starting transfere
(DD)[client.c@157]: Ready to read status wrote
(DD)[client.c@167]: Message decoded: left space 924
(DD)[client.c@167]: Message decoded: left space 848
(DD)[client.c@167]: Message decoded: left space 772
(DD)[client.c@167]: Message decoded: left space 696
(DD)[client.c@167]: Message decoded: left space 620
(DD)[client.c@167]: Message decoded: left space 544
(DD)[client.c@167]: Message decoded: left space 468
(DD)[client.c@167]: Message decoded: left space 392
(DD)[client.c@167]: Message decoded: left space 316
(DD)[client.c@167]: Message decoded: left space 240
(DD)[client.c@167]: Message decoded: left space 164
(DD)[client.c@167]: Message decoded: left space 88
(DD)[client.c@167]: Message decoded: left space 12
(DD)[client.c@167]: Message decoded: left space -64
(DD)[client.c@211]: Client decode finished
client process - OK
```

Результат автономной проверки

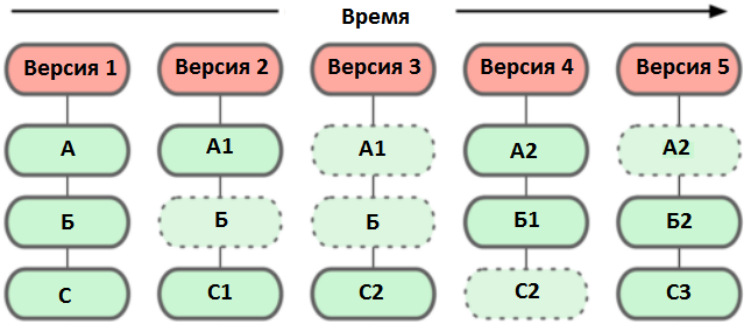
Технология разработки библиотеки

Инструменты разработки

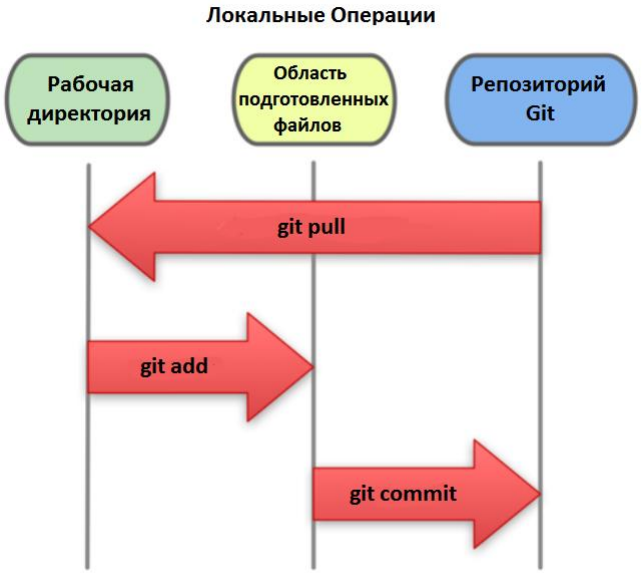
- **Git** - распределённая система контроля версий



- Слепки вместо патчей



- Три состояния файла

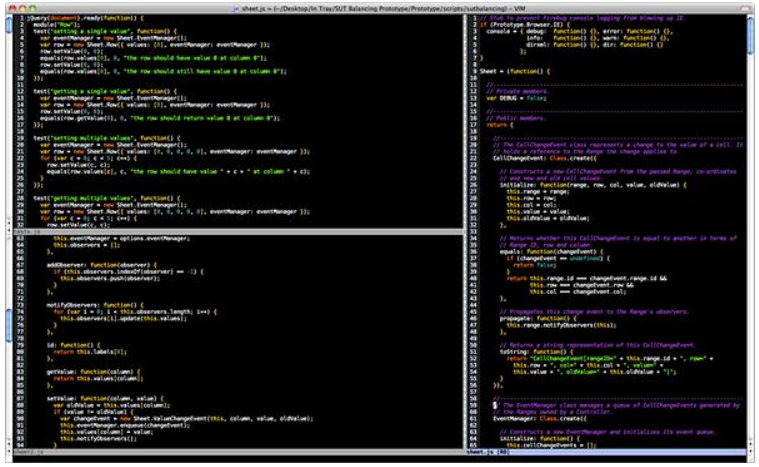


Технология разработки библиотеки

Инструменты разработки

Vim - текстовый редактор с двумя режимами ввода:

- Командный (позволяет использовать клавиши клавиатуры не для печати символов, а для различных команд)
- Текстовый (режим непосредственного редактирования текста, аналогичный большинству «обычных» редакторов).

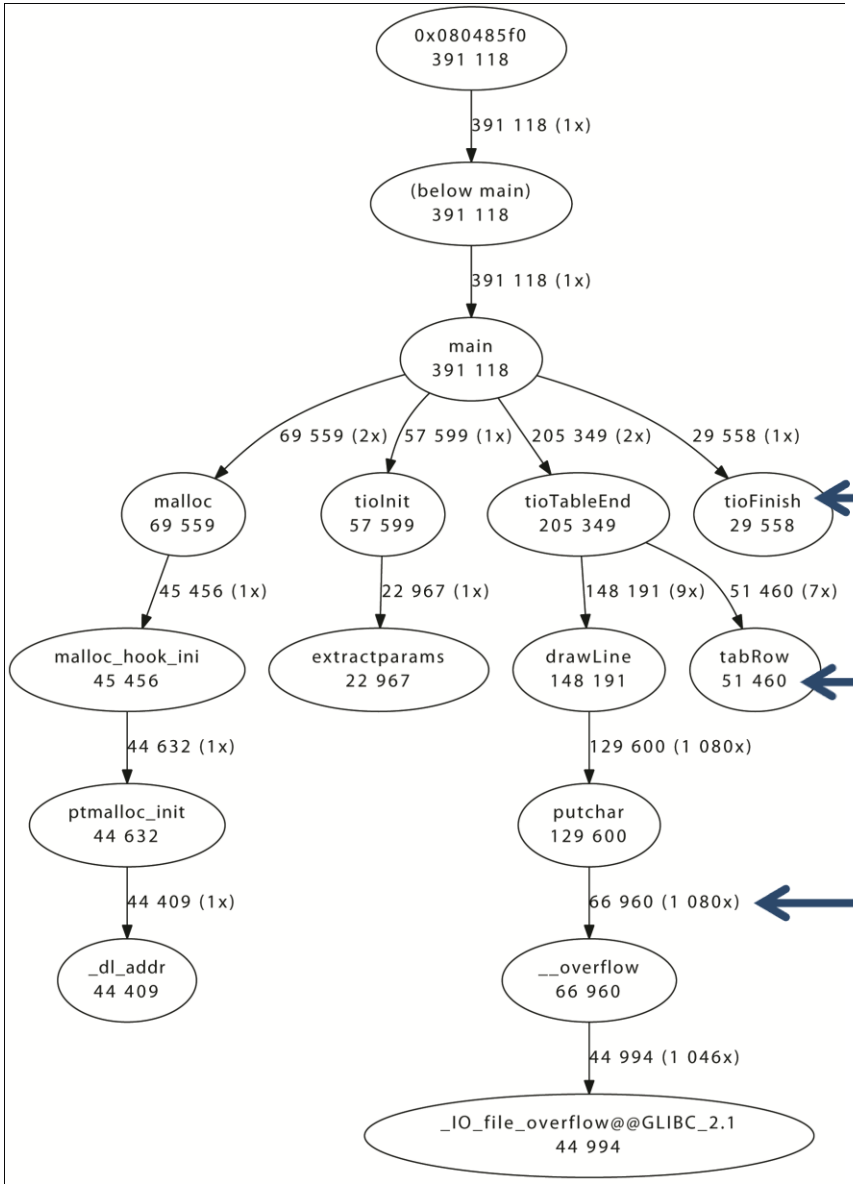


Функциональность

- Разбиение окон редактирования
- Неограниченная глубина отмены (undo)
- Режим сравнения двух файлов
- Подсветка синтаксиса
- Автоматическое продолжение команд
- Сворачивание (folding) текста
- Поддержка цикла разработки «редактирование — компиляция — исправление» программ

Технология разработки библиотеки

Профилирование



Используемый профилировщик:
Valgrind
Инструмент:
Callgrind
Визуальный инструмент для просмотра данных профилирования:
KCachegrind

Имя вызываемой функции

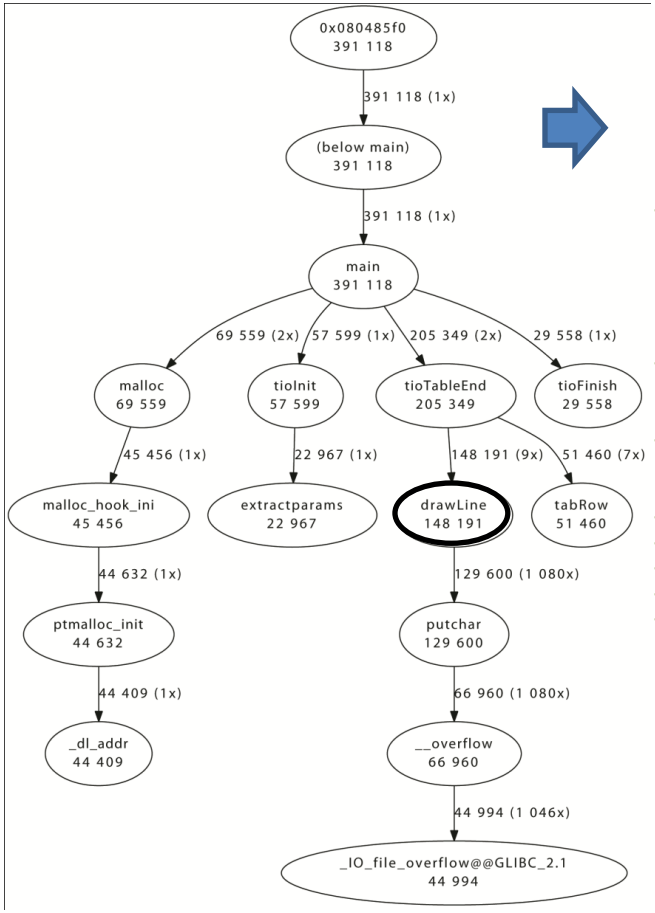
Число процессорных операций, потраченных на выполнение

Количество вызовов функций

Технология разработки библиотеки

Оптимизация по результатам профилировки

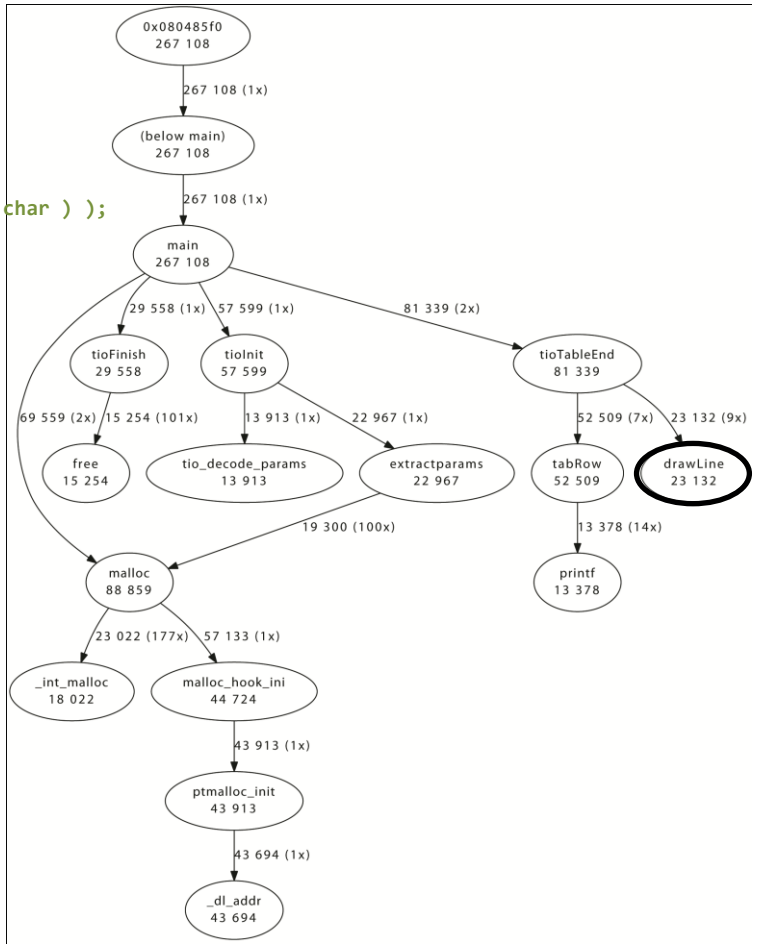
До



Внесенные изменения

```
int drawLine( int lenColCon )
 {

+    char *pLine = malloc( WIDTH * sizeof( char ) );
     int i;
     for( i = 0; i < WIDTH; ++ i )
     {
         if((i % lenColCon) == 0)
-            printf("+");
+            pLine[i] = '+';
         else
-            printf("-");
+            pLine[i] = '-';
     }
-    printf( "+\n" );
+    pLine[i] = '+';
+    pLine[++i] = '\n';
+    fputs( pLine, stdout );
+    free(pLine);
+    pLint = NULL;
     return 0;
 }
```

После



Спасибо за Внимание