1. ОБЗОР ЛИТЕРАТУРЫ

В настоящее время проблема превращения жёсткого диска компьютера в файловую свалку появляется практически у каждого пользователя. Минутное желание скачать файл, создание одноразовых каталогов без последующего удаления, копирование файлов с накопителей, нежелание создавать структуру директорий для своего жёсткого диска или нежелание этой структуры придерживаться – всё это приводит к тому, что жёсткий диск пользователя превращается в беспорядочное нагромождение файлов и папок, и нахождение нужной информации на диске превращается в нетривиальную задачу. В настоящем дипломном проекте рассматривается программное приложение, способное поддерживать первоначально созданную пользователем структуру каталогов без его участия, в автоматическом режиме.

Файлы могут попадать на жёсткий диск пользователя разными путями: скачивание файлов из интернета, обмен документами с помощью USB-накопителей, файлы, создаваемые приложениями во время работы, файлы, которые создаёт сам пользователь. Все эти файлы потенциально вызывают нарушение первоначальной структуры каталогов пользователя, особенно в течении большого количества времени. Что бы показать масштаб проблемы нужна статистика файлового обмена. Статистику по файловому обмену на USB носителях и созданию файлов пользователей найти невозможно, поскольку эта информация никем не собирается. Но статистика, позволяющая сказать насколько часто файлы качают из интернета - находится в открытом доступе.

Наибольшее количество файлов попадает на жёсткий диск пользователя из сети Internet. По статистике Google Trends запросы со словом «скачать» имеют стабильную популярность. Она колеблется в районе 75/100 и достигает пика во время новогодних праздников 90-100/100 (см. рисунок 1.1). Больше всего таких запросов приходится на страны СНГ. Наибольшая популярность – в Казахстане 89/100 (см. рисунок 1.2) [1]. По статистике Яндекс, поисковик ежемесячно выдаёт 210 628 770 ответов на запросы со словом «скачать» [2]. Анализ аналогичных запросов в англоязычных странах показывает, что пользователи из этих стран качают файлы чаще чем аналогичные пользователи из стран СНГ. По статистике того же Google Trends популярность запроса со словом «download» колеблется в районе 75-100/100 (см. рисунок 1.3). Наиболее популярны такие запросы в Индии, Пакистане и Индонезии (см. рисунок 1.4) [3].

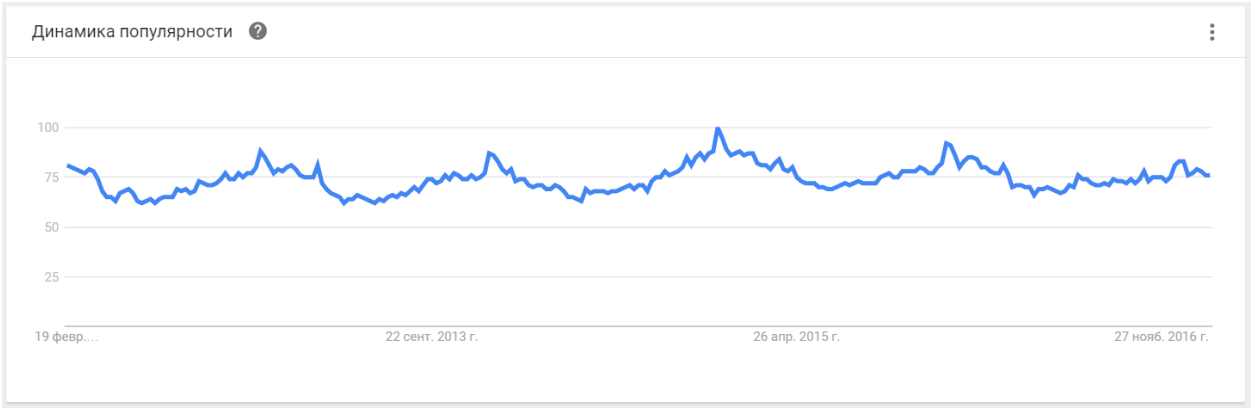

Рисунок 1.1 - Статистика популярности запросов со словом «скачать»

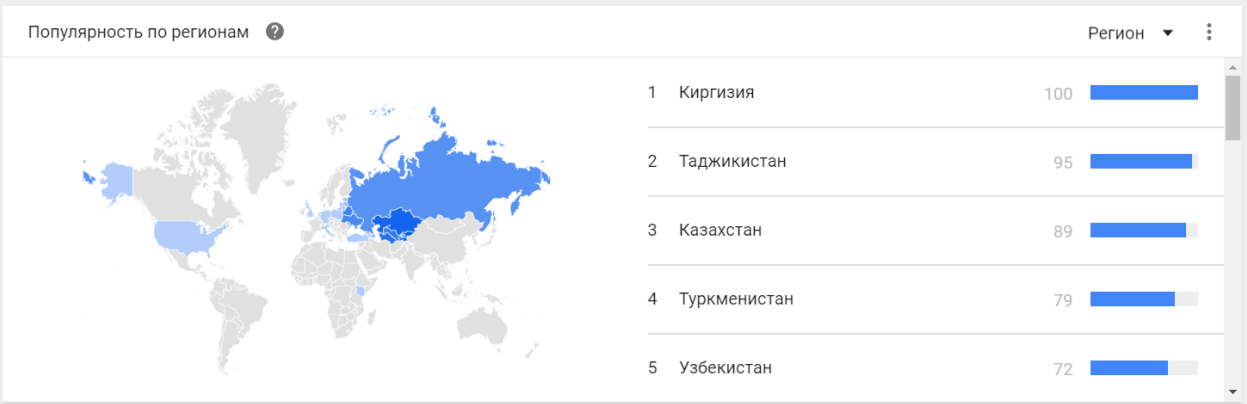

Рисунок 1.2 - Популярность запросов со словом «скачать» по регионам

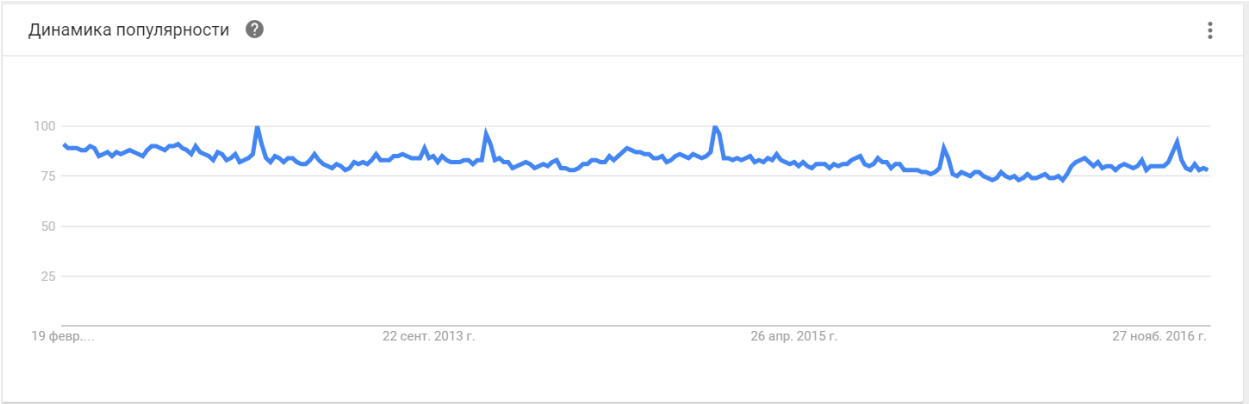

Рисунок 1.3 - Статистика популярности запросов со словом «download»

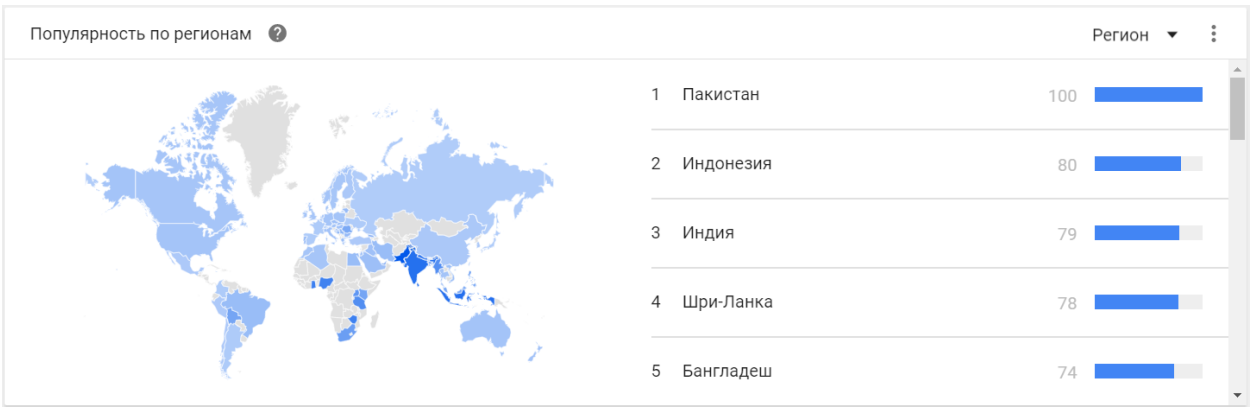

Рисунок 1.4 - Популярность запросов со словом «download» по регионам

Также об активности обмена файлами можно судить по активности использования торрент-клиентов по миру. Об этом можно судить по статистике, собранной компанией EZTV, специализирующейся на TV-торрентах (см. рисунок 1.5). В итоговой таблице собраны 357 миллионов уникальных пиров (равноправный участник одноранговой сети) по разным торрент-клиентам [4].

Rank	Client	Peer IDs
1	Xunlei	104,717,386 (29.308%)
2	µTorrent	92,086,035 (25.773%)
3	Azureus	86,055,354 (24.085%)
4	Mainline	17,207,544 (4.816%)
5	BitComet	14,341,918 (4.014%)
6	Transmission	11,637,110 (3.257%)
7	BitSpirit	7,132,119 (1.996%)
8	FlashGet	3,882,628 (1.087%)
9	TuoTu	2,531,472 (0.708%)
10	libtorrent	1,822,833 (0.51%)
11	Lphant	1,437,921 (0.402%)
12	BitComet	1,061,482 (0.297%)
13	BitTyrant	794,218 (0.222%)
14	Xtorrent	791,925 (0.222%)
15	rTorrent	702,498 (0.197%)
16	QQDownload	580,103 (0.162%)
17	Opera	556,902 (0.156%)
194	GetRight	2 (0%)

Рисунок 1.5 - Статистика использования торрент-клиентов по миру.

Как видно из статистики, обмен файлами сегодня проходит в огромных масштабах. Это значит, что типичному пользователю компьютера довольно часто приходится сталкиваться с проблемой поиска необходимых файлов на своём компьютере. Файлы, которые необходимо найти, потенциально попали на диск от дня до месяца назад, и были при этом сохранены в разные директории или сохранены в директорию, в которую скачиваются все без исключения файлы. Однократный поиск файла в такой ситуации может занять немного времени. При многократном поиске в большом скоплении файлов и папок, которое когда-то было структурой каталогов пользователя – будет затрачиваться значительное количество времени и сил.

Эта проблема существует уже давно. Время от времени появляются программы, позволяющие привести структуру каталогов пользователя в необходимый для него порядок. В программы, предназначенные для скачивания файлов, часто добавляются возможности, позволяющие этот порядок поддерживать.

По тому, насколько эффективно программа решает задачу поддержания в порядке структуры каталогов пользователя, программы, которые имеют такую функциональность, можно условно разделить на три категории:

1) Программы, позволяющие управлять сохранением файлов.
2) Программы, позволяющие разово упорядочить файлы в отдельно взятом каталоге пользователя.
3) Программы позволяющие долговременно сохранять структуру каталогов пользователя.

К программам первой категории можно отнести многочисленные менеджеры загрузки, торрент-клиенты, установщики программ, редакторы текста, музыки, видео и т.д. Эти программы, кроме своей основной функциональности, дают пользователю возможность управлять сохранением файлов, с которыми работает программа, тем самым позволяет пользователю сохранить его структуру каталогов. Сам механизм управления сохранением в таких программах может быть реализован по-разному: в одних программах файл можно сохранять разово, и с каждым новым файлом приходится заново выбирать путь, по которому он будет сохранён, в других программах путь можно указать в настройках. С другой стороны, эта же функциональность позволяет пользователю сохранять свои файлы куда попало, что производит обратный эффект – постепенное захламление жёсткого диска компьютера. В качестве примера таких программ можно привести всем известный менеджер загрузок μTorrent и текстовый редактор Notepad++ [5, 6].

К программам второй категории относятся многочисленные сортировщики файлов, программы, позволяющие удалить неиспользуемые временные файлы и программы, удаляющие файлы имеющие копии в разных каталогах.

Принцип работы с программами сортировщиками сводится к следующему – пользователь выбирает каталог, в котором он хочет отсортировать файлы, задаёт правила сортировки, которые отличаются от программы к программе, но в основном сводятся к указаниям куда переместить файлы определённого типа. Типичным примером программы-сортировщика файлов можно назвать DropIt [7]. Пример интерфейса программы DroptIt, в котором настраиваются правила сортировки можно увидеть на рисунке 1.6.

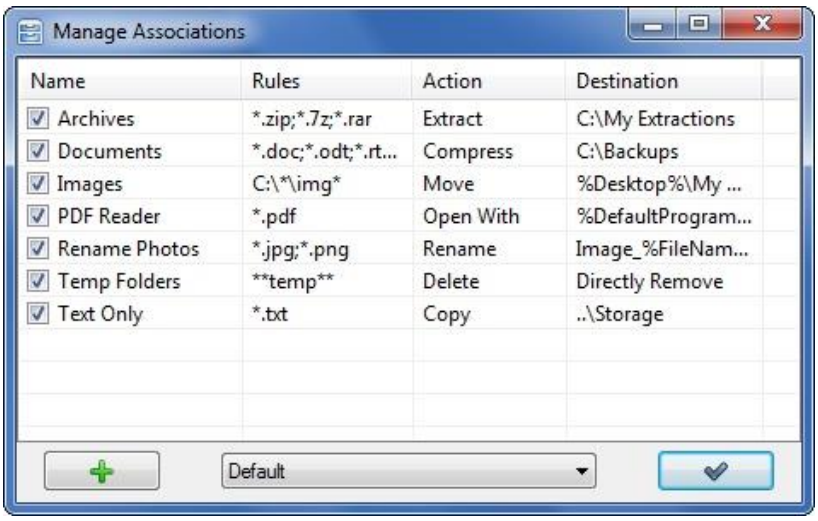

Рисунок 1.6 – Настройка правил сортировки в программе DropIt.

В качестве примера программы, позволяющей найти у удалить неиспользуемые временные файлы, файлы, хранящиеся в кэше браузеров, файлы, имеющие копии в других каталогах и так далее можно привести программу CCleaner [8]. Принцип работы с такими программами как CCleaner прост – пользователь отмечает что ему нужно, нажимает на кнопку, а программа дальше всё делает автоматически. Пример интерфейса окна поиска и удаления ненужных файлов на рисунке 1.7. Список программ, позволяющих искать и удалять файлы у которых есть копии в других директориях: DupKiller, DuplicateCleaner, CloneSpy, DuplicateFinder и т.д [9]. Пользователь задаёт какой-то критерий поиска (см. рисунок 1.8), программа ищет дубликаты, предлагает какой из N дубликатов удалить, и пользователь удаляет только то, что действительно должно быть удалено.

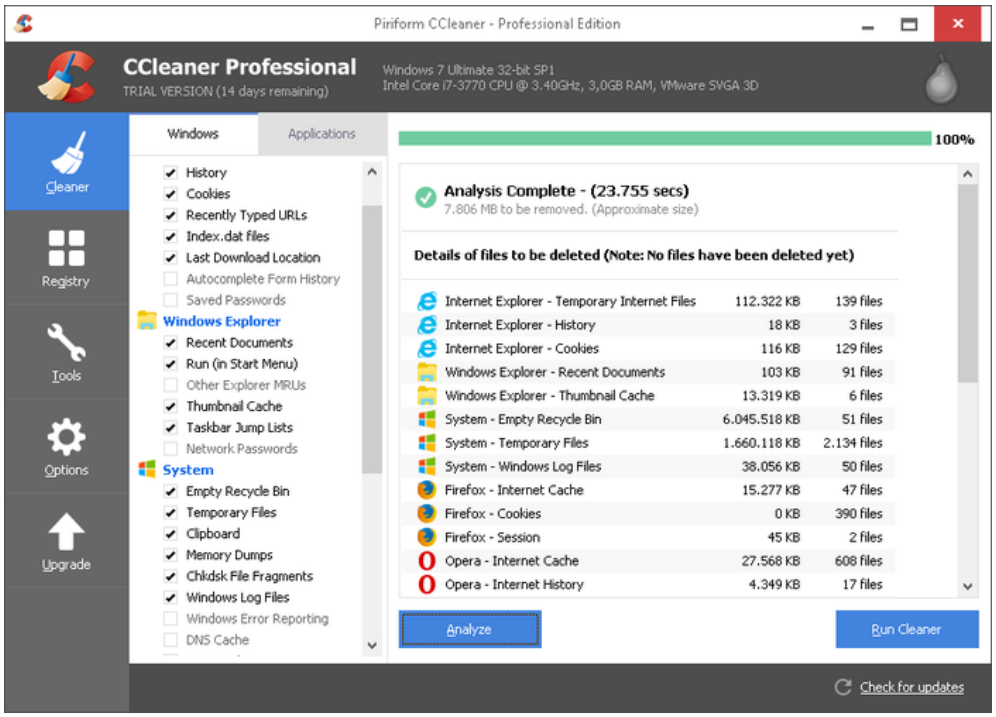

Рисунок 1.7 – Пример работы программы CCleaner

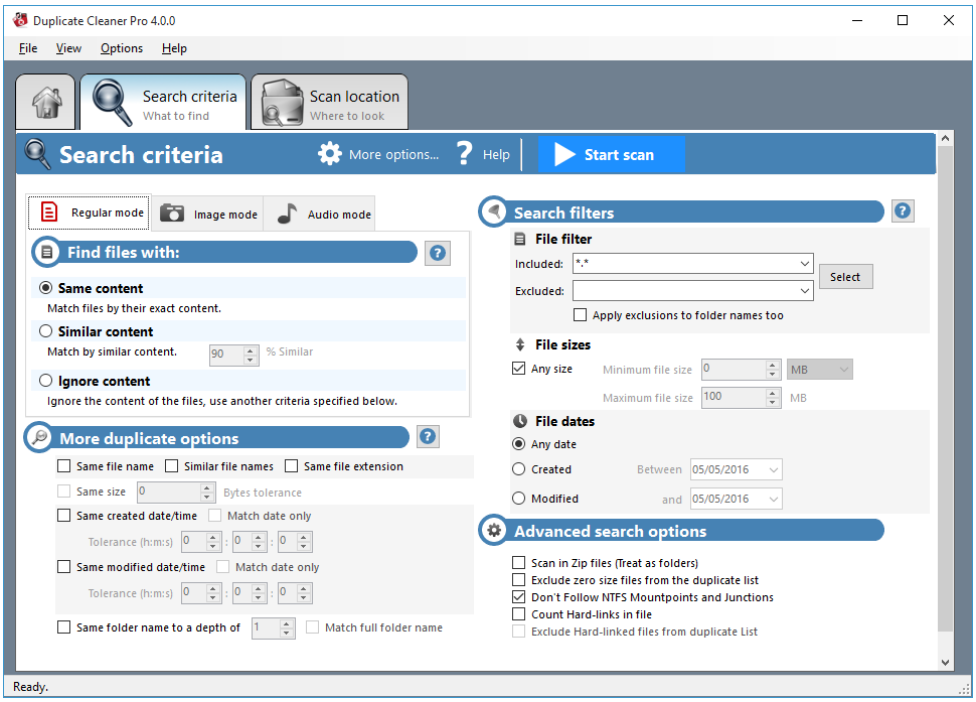

Рисунок 1.8 – Пример настройки критериев поиска в программе Duplicate Cleaner

Программы, относящиеся ко второй категории могут помочь пользователю довольно быстро восстановить первоначальную структуру его

каталогов. Минус таких программ в том, что они решают проблему воссоздания исходной структуры каталогов пользователя, а не её сохранения. Используют эти программы обычно, когда структура уже нарушена и не понятна пользователю, и нахождение нужных файлов начинает занимать много времени. Плюс же в том, что такие программы решают обычно какую-то одну задачу, а не пытаются делать всё сразу, поэтому, когда у пользователя появляется конкретная проблема, ему не нужно тратить часы что бы научится пользоваться этой программой – пользователь её запускает, программа отрабатывает, и он может спокойно работать дальше.

Если программы, подпадающие под первую категорию, имеют функциональность, позволяющую пользователю поддерживать порядок на рабочем диске, программы второй категории помогают этот порядок восстанавливать, то программы третьей категории умеют этот порядок поддерживать автоматически. На данный момент среди таких программ чаще всего встречаются решения, созданные программистами для своих собственный нужд и не получившие широкое распространение. Исключением является только программа File2Folder, которая работает только на OS X, и доступна для скачивания в Mac App Store [10], далее она будет рассмотрена в качестве аналога данного дипломного проекта. Ещё одним примером может служить программа Automator, которая создана как попытка автоматизировать сортировку папки загрузок, и работает только для OS X [11].

Рассмотрим подробнее программу File2Folder. Согласно описанию, размещённому на Mac App Store, программа умеет перемещать, копировать и удалять файлы по расширениям и части имени и переименовывать файлы согласно правилам, но в отличие от программ-сортировщиков есть возможность настроить таймер, который через заданный промежуток времени будет проверять заданные пользователем директории и распределять новые файлы в соответствии с заданными правилами.

Автоматическое выполнение каких-то действий через определённые промежутки времени – идея не новая, таким образом управляются многие задачи, которые требуют много ресурсов и времени, например, выполнение резервного копирования данных, сборка мусора в приложениях с автоматическим управлением памятью, обработка данных поступивших в базу данных за день. Но в задаче автоматического распределения файлов у этого подхода есть свои плюсы и минусы.

К плюсам подхода можно отнести простоту реализации, гибкость в управлении таймерами, которая позволяет настроить короткие промежутки,

для часто используемых директорий и длинные, для директорий, которые просто нужно изредка чистить. Так же к плюсам можно отнести интуитивную понятность подхода для пользователя.

К минусам же можно отнести то, что пользователю нужно дожидаться срабатывания таймера, чтобы увидеть файл в нужной директории, что не всегда удобно. К примеру, в ситуации, когда пользователь качает музыку, для того что бы скинуть её на плеер, по окончанию закачки файлов пользователь будет ожидать что вся музыка уже распределена по папкам, но в случае, когда таймер срабатывает раз в 5 минут – это будет не так, так как половина музыки может быть распределена, а другая половина дожидаться следующего таймера. Второй минус в том, что происходит большое количество холостых срабатываний программы если в директорию не добавляется никаких файлов, что не позволяет выставить очень короткий промежуток срабатывания таймера даже для часто используемых директорий. Так же это повышает требования к экономии программой ресурсов компьютера, так как частое выполнение проверок программой, без оптимизации программы, может привести к «зависанию» компьютера пользователя, что само-собой неприемлемо.

Альтернативой подходу с таймерами может стать подход с использованием механизмов файловой системы позволяющих отслеживать события, которые в ней происходят. К инструментам, позволяющим отслеживать такие события можно отнести библиотеку inotify в ОС Linux и OS X [12] и часть библиотеки WIN64 API в ОС Windows [13].

Плюсы подхода с отслеживанием событий в том, что нет лишних срабатываний и приложение будет отрабатывать сразу после того как файлы попали на диск. То есть такой подход компенсирует все минусы подхода с таймерами.

Что касается минусов – появляется проблема, связанная со срабатываниями событий. Дело в том, что то, что файл был создан в файловой системе ещё не значит, что все данные для этого файла полностью скопированы. В таком случае начало сортировки до полного копирования данных может привести к непредвиденным ошибкам и повреждённым данным, что недопустимо. Поэтому в рамках дипломного проекта необходимо создать механизм, способный эту ситуацию обрабатывать. Второй минус заключается в том, что автоматическая сортировка файлов сразу после попадания их на диск, увеличивает количество операций с диском, которых будет и так много, если файлы копируются, например, с внешнего накопителя.

Если процесс копирования и процесс сортировки будут пытаться оперировать информацией большей чем пропускная способность диска – это приведёт к существенному замедлению обоих процессов. Поэтому для того, чтобы эффективно использовать диск, лучше использовать комбинированный подход, и использовать события и таймеры в зависимости от целевого предназначения директории.

Кроме задачи автоматического отслеживания файлов, и задачи абстрагирования этого отслеживания от операционной системы, в рамках данного дипломного проекта необходимо решить задачу создания абстракции над файловой системой, для предоставления пользователю списка команд, предназначенных для управления отслеживанием и распределением файлов. Наличие таких команд позволит создать единообразный интерфейс (API) для разных операционных систем, который будет использоваться на самом высоком уровне данного дипломного проекта, и который позволит пользователям создавать в будущем свои программы на его основе. Примером такой абстракции с набором команд к ней, может служить абстракция репозитория, созданная Линусом Торвальдсом в программе git [14].

Для уровня отслеживания и распределения файлов в данном дипломном проекте используется язык C, который хорошо подходит для обеспечения быстродействия и обеспечивает доступ к низкоуровневым функциям операционных систем [15].

На уровне API используется язык Python, для обеспечения переносимости между разными операционными системами и средами. Для обеспечения работы этого уровня в определённом окружении нужно будет только установить интерпретатор и нужные библиотеки [16].

Для уровня пользовательского интерфейса могут быть использованы разные технологии, которые будут зависеть от операционной системы, это может быть, как консоль в ОС Linux, так и WPF приложение в ОС Windows. При наличии полного API выбор технологии будет вестись по таким параметрам как простота создания интерфейса для программиста и его удобство для конечного пользователя.