4 ПРОЕКТИРОВАНИЕ ФУНКЦИОНАЛЬНЫХ ВОЗМОЖНОСТЕЙ ПРОГРАММЫ

4.1 Функции программного обеспечения

Основной функцией разработанного в рамках курсового проекта программного обеспечения является диагностика и оптимизация энергопотребления. Данный функционал приложения достигается путём использования встроенных в Windows схем управления питанием и анализа текущих данных о батарее, установленной в устройстве. Помимо этого, программа выполняет функцию монитора использования системных и аппаратных ресурсов, предоставляя пользователю информацию о загрузке ядер (потоков) центрального процессора, о количестве физических ядер центрального процессора, о количестве виртуальных ядер (потоков) центрального процессора, об общей нагрузке на центральный процессор, о максимальной и минимальной частотах работы процессора, об общем объёме оперативной памяти устройства, о занятом объёме оперативной памяти, о загрузке накопителя данных по операциям чтения и записи, об общей загрузке графического процессора, об общем объёме оперативной памяти графического процессора, о занятом объёме оперативной памяти графического процессора, о запасе батареи в процентном и временном выражениях и статусе батареи устройства. Предоставляемый детальный мониторинг ресурсов аппаратных и системных ресурсов даёт пользователю возможность наблюдать изменение использования этих ресурсов в зависимости от установленной схемы управления питания.

4.2 Методы получения информации об использовании ресурсов

Получение информации о ресурсах и их потреблении осуществляется с помощью библиотек psutil и GPUtil.

Модуль psutil – это кроссплатформенная библиотека для получения информации о запущенных процессах и использовании системы (ЦП, память, диски, сеть, датчики) в Python. Этот модуль полезен для мониторинга системы, профилирования, ограничения ресурсов процессов и управления запущенными процессами [13].

Информация о загруженности ядер (потоков) получается с помощью использования psutil.cpu_percent(). Функция psutil.cpu_percent() возвращает число float, представляющее текущую загрузку ЦП в масштабе всей системы в процентах. Когда аргумент interval больше 0, то сравнивается время системного процессора, прошедшее до и после интервала (блокировка). Когда интервал равен 0 или None, то сравнивается время системного процессора, прошедшее с момента последнего вызова или импорта модуля, с немедленным возвратом значения. Это означает, что при первом вызове он вернет

бессмысленное значение 0, которое необходимо игнорировать. В этом случае для точности рекомендуется вызывать эту функцию с интервалом не менее 0.1 секунды между вызовами. Когда аргумент percpu имеет значение True, то функция возвращает список чисел float, представляющий использование в процентах для каждого ЦП. Первый элемент списка относится к первому процессору, второй элемент – ко второму процессору и так далее. Порядок списка одинаков для всех вызовов [14].

Информация о количестве ядер и потоков центрального процессора предоставляется методом psutil.cpu_count(). Функция psutil.cpu_count() возвращает количество логических процессоров в системе (так же, как os.cpu_count) или None, если оно не определено. «Логические процессоры» означают количество физических ядер, умноженное на количество потоков, которые могут выполняться на каждом ядре (Hyper-Threading). Если для аргумента logical задано значение False, то возвращает только количество физических ядер или None, если оно не определено [14].

Информация о минимальной и максимальной частоте работы центрального процессора получается использованием функции psutil.cpu_freq(). Функция psutil.cpu_freq() возвращает частоту ЦП в виде именованного кортежа, включая текущую, минимальную и максимальную частоты, выраженные в МГц [14].

Путём использования метода psutil.virtual_memory() получается информация об оперативной памяти. Функция psutil.virtual_memory() возвращает статистику об использовании системной памяти в виде именованного кортежа, включающего следующие поля, выраженные в байтах. Основные показатели: total (общая физическая память без подкачки SWAP), available (память, которую можно мгновенно отдать процессам без перехода системы в SWAP. Значение вычисляется путем суммирования различных значений памяти в зависимости от платформы, и предполагается, что оно будет использоваться для мониторинга фактического использования памяти кроссплатформенным способом), used (используемая память, рассчитанная по-разному в зависимости от платформы и предназначена исключительно для информационных целей) [15].

Информацию о дисковой нагрузке предоставляет метод psutil.disk_io_counters(). Функция psutil.disk_io_counters() возвращает общесистемную статистику дискового ввода-вывода в виде именованного кортежа, включающего следующие поля: read_count (количество считываний), write_count (количество записей), read_bytes (количество прочитанных байт), write_bytes (количество записанных байт) [16].

GPUtil – это модуль Python, который предоставляет интерфейс для получения информации о графических процессорах (GPU) и их загрузке. Он полезен для разработчиков, которые работают с приложениями, использующими ускорение GPU, особенно в области машинного обучения и глубокого обучения.

Вся информация о графическом процессоре предоставляется методом GPUtil.getGPUs(). getGPUs() – это функция, которая возвращает список доступных графических процессоров (GPU) в системе. Она является частью модуля GPUtil, который предоставляет интерфейс для работы с GPU. Функция getGPUs() возвращает список объектов GPU, каждый из которых содержит информацию о соответствующем GPU, включая идентификатор, имя, загрузку, свободную память, используемую память и общую память о соответствующем графическом процессоре.

Информация о батарее, как и информация о центральном процессоре и оперативной памяти, предоставляется модулем psutil, а конкретно – методом psutil.sensors_battery(). Функция psutil.sensors_battery() возвращает информацию о состоянии батареи в виде именованного кортежа, включая количество процентов запаса батареи, статус (заряжается или разряжается) и количество оставшихся секунд работы компьютера от батареи. Если батарея не установлена или метрики не могут быть определены, возвращается None [17].

Вывод информации об использовании аппаратных и системных ресурсов реализован с помощью графических компонентов вывода Tkinter.

Для отображения загруженности ядер (потоков) используется Treeview. Виджет Treeview предназначен для отображения иерархических данных, причем как в виде дерева, так и в виде таблицы. Среди параметров Treeview принято выделять columns (столбцы таблицы в виде строки или списка или кортежа строк), displaycolumns (отображаемые столбцы таблицы), cursor (курсор при наведении на виджет), height (высота виджета), padding (отступы от границ виджета до содержимого), selectmode (режим выбора элементов в виджете), show (формат отображения данных). Для добавления данных применяется метод insert(), для удаления данных – метод delete(), для перемещения элемента на другую позицию – метод move(), для получения элементов – метод get_children(), если надо изменить один столбец, то применяется метод set() [18].

Все остальные показатели использования ресурсов выводятся с помощью Label. Виджет Label представляет текстовую метку. Этот элемент позволяет выводить статический текст без возможности редактирования [19].

4.3 Политика изменения схемы управления питанием

Разработанный программный продукт выполняет функции диагностики и оптимизации энергопотребления, скрывая принцип выбора схемы управления питанием от пользователя. Ключевыми показателями при выборе подходящей схемы являются статус батареи компьютера и ее запас в процентном выражении.

Windows предоставляет три схемы управления питанием: энергосберегающая схема, сбалансированная схема и схема максимальной производительности. Их принципиальная разница заключается в изменении

политики использования системных и аппаратных ресурсов, изменении необходимого времени бездействия для отключения экрана и перехода компьютера в режим сна и регулировке яркости. Для доступа к данным схемам используются их GUID (global unique identifier) – глобальный уникальный идентификатор. На каждом компьютере с Windows эти три схемы имеют одинаковые GUID, что гарантирует возможность использования разработанной программы на любом из компьютеров с установленной Windows.

После своего запуска программа собирает данные об использовании системных и аппаратных ресурсов, обновляя их каждые полторы секунды. После того, как программа получила первичную информацию о ресурсах, пользователю становится доступна диагностика энергопотребления. Под диагностикой энергопотребления понимается проверка текущей схемы управления питанием на идентичность с рекомендуемой. Рекомендуемая схема управления питанием устанавливается в зависимости от статуса батареи и ее запаса в процентном выражении.

Если батарея заряжается, то есть компьютер работает от сети, то рекомендуемой схемой управления питанием считается схема максимальной производительности.

В случае, когда батарея разряжается, учитывается непосредственно запас батареи в процентном выражении. При разрядке батареи и нахождении ее запаса в промежутке от 50 до 100 процентов рекомендуемой схемой управления питанием принимается сбалансированная схема. Если же запас батареи находится в промежутке от 0 до 49 процентов и, она разряжается, то в качестве рекомендуемой схемы устанавливается энергосберегающая схема.

После нажатия кнопки диагностики проверка соответствия схем управления питанием устанавливает переменную is_optimization_needed в False, если установлена рекомендуемая схема, и в True, если установленная на текущий момент схема не идентична рекомендуемой схеме.

В случае, когда переменная is_optimization_needed установлена в True, становится активной кнопка оптимизации, при нажатии на которую текущая схема управления питанием изменяется на рекомендуемую, после чего кнопка становится неактивной.

Для непосредственного изменения и получения текущей схемы управления питанием используется модуль subprocess. Модуль subprocess отвечает за выполнение следующих действий: порождение новых процессов, соединение с потоками стандартного ввода, стандартного вывода, стандартного вывода сообщений об ошибках и получение кодов возврата от этих процессов [20]. С помощью функции run() из этого модуля создаётся процесс PowerShell, который выполняет команду powercfg /getactivescheme для получения текущей схемы управления питанием и powercfg /setactive {power_scheme_guid} для установки рекомендуемой схемы. Здесь под {power_scheme_guid} понимается GUID одной из предоставляемых Windows схем управления питанием.