**Спецификация требований**

**программного обеспечения**

(**Software Requirements Specification)**

**ЧУСИК™:**

Чудо-сборщик информации

Компьютера

Маслов Роман

Морев Максим

Стасьев Денис

Гришнов Илья

Новиков Юрий

Долгополов Иван

Пахомов Михаил

Структура документа

1. **Вступление**
   1. Цель
   2. Возможности
   3. Аудитория
   4. Термины
   5. Источники
2. **Общее описание**
   1. Виденье
   2. Общая функциональность
   3. Классы пользователей
   4. Ограничения
3. **Специальные требования**
   1. Пользовательский интерфейс
   2. Специальные функциональные требования
4. **Приоритеты работ и план релиза**
   1. Приоритеты
   2. Временные рамки разработки
5. **Внешние интерфейсы**
   1. Программные интерфейсы
   2. Интерфейсы оборудования
   3. UI/UX
6. **Вступление**

Этот раздел дает краткое описание возможностей всего того, что будет представлено в данном документе.

* 1. **Цель**

Целю настоящего документа является подробное описание требований для программного обеспечения «Чудо — сборщика информации компьютера» (ЧУСИК™). Документ продемонстрирует функционал и полное описание процесса разработки данного проекта. Так же он объяснит ограничения и возможности графического интерфейса.

**1.2 Возможности**

« Чудо — сборщик информации компьютера» - это ПО, способное удаленно получать информацию о производительности компьютера, а именно : CPU , оперативная память.

Более того, программа способна получать данные сразу с нескольких компьютеров, тем самым собирая данные о производительности машин в локальной сети.

**1.3 Аудитория**

Данная программа может быть востребована для людей, которые желают посмотреть информацию о производительности отдельного компьютера и/или производительности целой локальной сети компьютеров.

**1.4 Термины**

Клиент – человек, с машины которого будут сниматься показатели производительности.

Администратор – человек, обладающей утилитой управления.

Утилита управления – часть программы, установленная на компьютере Администратора, и способная задавать настройки для снятия показаний с остальных машин сети.

Агент – часть программы, установленная на машинах клиентов.

Объект – компьютер, с которого снимаются показатели производительности.

**1.5 Источники**

Интернет и курс “Жизненный цикл ПО”.

**2. Общее описание**

Этот раздел предоставит общее описание проекта, а именно:

1. Отобразит основной функционал продукта
2. Распределит пользователей по взаимодействующим классам
3. Покажет ограничения при работе
4. Даст общее видение работы системы

**2.1 Видение**

Система будет состоять из двух частей:

1. Утилита управления
2. Агент

Агент устанавливается на подконтрольные машины, а утилита управления – на компьютер Администратора. Агенты должны периодически собирать информацию о производительности (CPU, RAM) и отправлять её на компьютер Администратора. Передача данных происходит по сети.

**2.2 Общая функциональность**

1. Возможность вывести список запущенных процессов. В этом списке для каждого процесса указывается его имя, потребляемые им ресурсы(CPU,RAM), а также имя пользователя, от имени которого данный процесс запущен.

2. С помощью утилиты управления задаются настройки, меняющие список обслуживаемых компьютеров. Там же просматриваются собранные данные.

3. Имеется функция изменения периода сбора информации с компьютеров.

4. Имеется функция выбора собираемой и выводимой на экран информации о компьютерах.

5. Все действия по изменению настроек и состава обслуживаемой сети компьютеров записываются в журнал.

6. История собираемой информации сохраняется в единую базу данных.

**2.3 Классы пользователей**

Существует два класса (типа) пользователей, взаимодействующих с данным программным обеспечением:

Пользователь агента (он же Клиент) – человек, на компьютере которого установлен агент.

Пользователь утилиты управления (он же Администратор) – человек, который имеет утилиту управления, посредством которой он взаимодействует с получаемой информацией о производительности подконтрольных компьютеров. Администратор задаёт настройки частоты приёма (периодичности приёма) данных, может сохранять получаемую информацию, а также менять список компьютеров, с которых производится сбор данных.

**2.3 Ограничения**

1. Максимальное количество контролируемых компьютеров – 10.

(Чтобы не перегружать сеть)

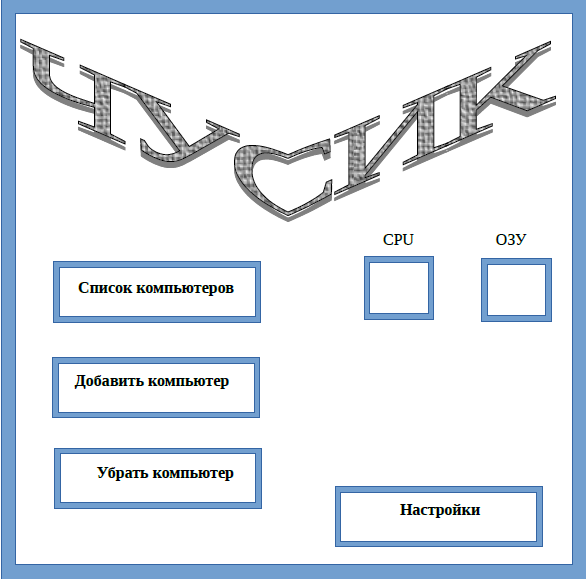
2. Минимальное количество контролируемых компьютеров – 1;

**3. Специальные требования**

Этот раздел содержит все специальные функциональные требования, а также требования к пользовательскому интерфейсу программы.

**3.1 Пользовательский интерфейс**

Интерфейс утилиты управления:



**3.2 Специальные функциональные требования**

Этот раздел содержит …

-\\- -\\--\\- -\\--\\- -\\--\\- -\\--\\- -\\--\\- -\\--\\- -\\--\\- -\\--\\- -\\--\\- -\\--\\- -\\-

(Ожидает уточнения в дальнейшем)

**4. Приоритеты работ и план релиза**

**4.1 Приоритеты**

1) Создание системы удаленного мониторинга производительности компьютеров. То есть создание клиентской и серверной частей только с простейшими функциями.

2) Создание возможности вывода списка запущенных на подконтрольных компьютерах процессов, с информацией об именах этих процессов, потребляемыми ими ресурсами (CPU, RAM), а также имени пользователя, от прав которого эти процессы запущены.

3) Создания возможности изменения настроек программы в ходе её работы.

4) Реализация журнала.

5) Написание графического интерфейса.

6) Организация базы данных.

7) Кроссплатформенность.

**4.2 Временные рамки разработки**

До конца нынешнего семестра.

**5. Внешние интерфейсы**

В данном пункте вводится описание программных и физических интерфейсов, а также строится модель взаимодействия частей системы как единого целого (UI/UX design).

**5.1 UI/UX design**

В сети, работающей по стеку протоколов TCP/IP, будем пользоваться понятием сокета – совокупности IP адреса и порта.

Управляющая утилита будет слушать один конкретный порт (условно порт A), на который будет приходить информация от обслуживаемых машин. При этом управляющая утилита будет использовать некий порт B, с помощью которого можно отправлять команды для настройки агентов на контролируемых компьютерах.

На стороне агентов по аналогии будем использовать два порта – один для приёма команд на изменение настроек (условно B` ), а другой для отправки данных о производительности (назовём A`).

Кратко и схематично:

