Разработка серверного и консольного клиентского приложения для обмена сообщения с ним для QNX в среде QNX Mnemonics IDE

Лабораторная работа

Ревизия: 0.1

Последняя модификация: Vlad Kovtun

Дата последней модификации: 25.11.2010 15:43:00

История изменений

24.11.2010 – Версия 0.1. Первичный документ. Влад Ковтун

Содержание

Лабораторная работа 14. Разработка серверного и консольного клиентского приложения для обмена сообщения с ним для QNX в среде QNX Mnemonics IDE

Вопросы

- Постановка задачи.
- Методические рекомендации.

Постановка задачи

Цель

1. Разработать серверное приложение.

2. Разработать клиентское приложение для обмена сообщения пользователя с сервером посредством протокола TCP/UDP.

Задачи

1. Ознакомиться с особенностями построения серверного приложения для ОС QNX.

2. Ознакомиться с особенностями настройки серверного сокета в ОС QNX.

3. Ознакомиться с особенностями межсетевого взаимодействия в ОС QNX посредством сокетов по протоколам TCP/UDP.

4. Разработать сервер для ОС QNX для обмена текстовыми сообщениями – ведения текстовых логов с клиентского консольного приложения.

4.1. В текстовом файле в формате UTF-8 необходимо фиксировать следующую информацию:

- IP адрес, с которого передано сообщение.
- Дата и время получения сообщения.
- Имя пользователя, который запустил клиентское приложение.
- Марка процессора, на котором выполняется клиентское приложение.
- Текст самого сообщения.

5. Разработать клиентское консольное приложение для ОС QNX, которое взаимодействует с сервером логирования посредством сокетов по протоколам TCP/UDP.

6. Оценить производительность сервера (какое количество записей фиксированной длины он сможет фиксировать в секунду).

7. Оформить отчет к лабораторной работе. В отчете обязательно указать, каким образом производится тестирование на корректность.

Задание

Управление сервером осуществляется посредством следующих команд:

- -lf:<logfilepath> – полный путь и имя файла лога, куда будет производиться запись сообщений.
- -p:<pidfilepath> - полный путь и имя файла, который будет использоваться для синхронизации процессов – сервера, что позволит запустить лишь единственную версию демона.
- -addr:<IP address> - IP адрес серверного сокета, обязательный параметр.
- -port:<port> - порт серверного сокета, обязательный параметр.
- -conn:<connections count> - максимальное количество одновременных подключений к серверному сокету, по умолчанию количество подключений равно 7.
- -s:<on|off> - выводить статус получения и обработки полученных от клентов сообщений на экран.
- -start – запустить на выполнение сервер.
- -stop – остановить функционирующий сервер.
- -restart – перезапустить сервер.
- -? - вывод информации о допустимых ключах командной строки.

Управление клиентским консольным приложением осуществляется посредством команд:

- -addr:<IP address> - IP адрес сервера, к которому следует подключиться. Обязательный параметр.
- -port:<port> - порт серверного сокета, к которому следует подключиться. Обязательный параметр.
- -m:<message> - текст сообщения, которое следует занести в лог-файл.
- -? - вывод информации о допустимых ключах командной строки.

Пример

Пример запуска демона, который хранит принятые сообщения в файле `log.txt`, а файл синхронизации называется `logger.pid`, кроме того используется один из локальных сетевых интерфейсов сервера с адресом `192.168.0.1` и портом `12345`:

```
[vladk@qnx-station]->loggerd -addr:192.168.0.1 -port:12345
 -if:tmp\logs\log.txt -p:logger.pid
```

Пример запуска клиентского консольного приложения с внесением в лог-файл сообщения «hello world!», осуществляется подключение к серверу по адресу `192.168.0.1` к порту `12345`:

```
[vladk@ qnx-station]->loggerc -addr:192.168.0.1 -port:12345 -m:hello world!
```

Вывод

Во время работы приложения, рекомендуется выводить информацию о статусе приложения, а также о корректности его работы на консоль. При получении сообщения от клиента и корректной его обработке, серверу следует информировать клиентов о статусе обработки сообщения.

Клиентское приложение выводит информацию о статусе обработки сообщения сервером, на консоль.

Серверное приложение выводит полученные сообщения от клиентов на консоль, с указанием адреса, порта, даты и времени получения сообщения.

Требования

- Архитектура приложения строится по модульному принципу.
- За основу принимается стандартная библиотека C++ (в случае разработки на C++).
- Рекомендуется использовать защищенные ресурсы и указатели.
- Обязательным является обработка исключений.
- Исходный код обязан быть комментирован. Для C++ следует использовать нотацию doxygen [5].

Тестирование

Тестирование приложения осуществляется в несколько этапов:

- Воспользоваться тестовыми пакетными файлами для проверки корректности функционирования.

Методические указания для самостоятельной работы

Теоретические сведения

Информация по работе с QNX Momentics IDE, доступна [4].

При разработке программного обеспечения в рамках лабораторной работы, рекомендуется воспользоваться готовыми библиотеками классов, из приведенных в [6, 7, 8].

Литература

1. Д. Алексеев, Е. Ведричев, А. Волков и др. Практика работы с QNX. –М: Издательский дом «КомБук», 2004. -432 с.

2. Роб Кертен. Введение в QNX/Neutrino 2. Руководство по программированию приложений реального времени в QNX Realtime Platform. –СПб.: Издательство «Петрополис», 2001. -480 с.

3. QNX official web-site URL: http://qnx.com

4. QNX Momentics - IDE User's Guide. URL: http://www.qnx.com/download/feature.html?programid=10404

5. Формирование документации к исходному коду с помощью средства doxygen. URL: www.nrjetix.com/r-and-d/lectures

6. Single Server With Multiple Clients: a Simple C++ Implementation. URL: http://www.codeproject.com/KB/IP/singleServerMulClient.aspx

7. Multi-threaded Client/Server Socket Class. URL: http://www.codeproject.com/KB/IP/serversocket.aspx

8. A light-weighted client/server socket class in C++. URL: http://www.codeproject.com/KB/IP/client_server_socket.aspx