Пермский филиал федерального государственного автономного образовательного учреждения высшего образования   
«Национальный исследовательский университет   
«Высшая школа экономики»

*Факультет экономики, менеджмента и бизнес-информатики*

Чепоков Елизар Сергеевич

**MSMQ**

*отчет по практическому заданию №4*

студента образовательной программы «Программная инженерия»

по направлению подготовки *09.03.04 Программная инженерия*

**Руководитель:**

доцент кафедры

информационных

технологий в бизнесе

Е. Б. Замятина

Пермь, 2021 год

**Содержание**

[Задание 1 3](#_Toc95958241)

[Задание 2 4](#_Toc95958242)

[Задание 3 5](#_Toc95958243)

[Задание 4 6](#_Toc95958244)

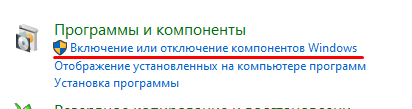
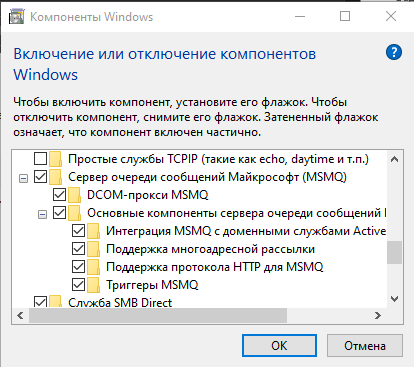
[Задание 5 6](#_Toc95958245)

[Задание 6 7](#_Toc95958246)

# Задание 1

Убедитесь, что у вас установлена служба очередей сообщений. Для этого в «Панели управления» выберите раздел «Программы и компоненты» выберите пункт «Включение и отключение компонентов Windows». В открывшемся дереве найдите вершину «Сервер очереди сообщений Майкрософт». Если данный компонент не установлен, то доустановите его.

**Решение**

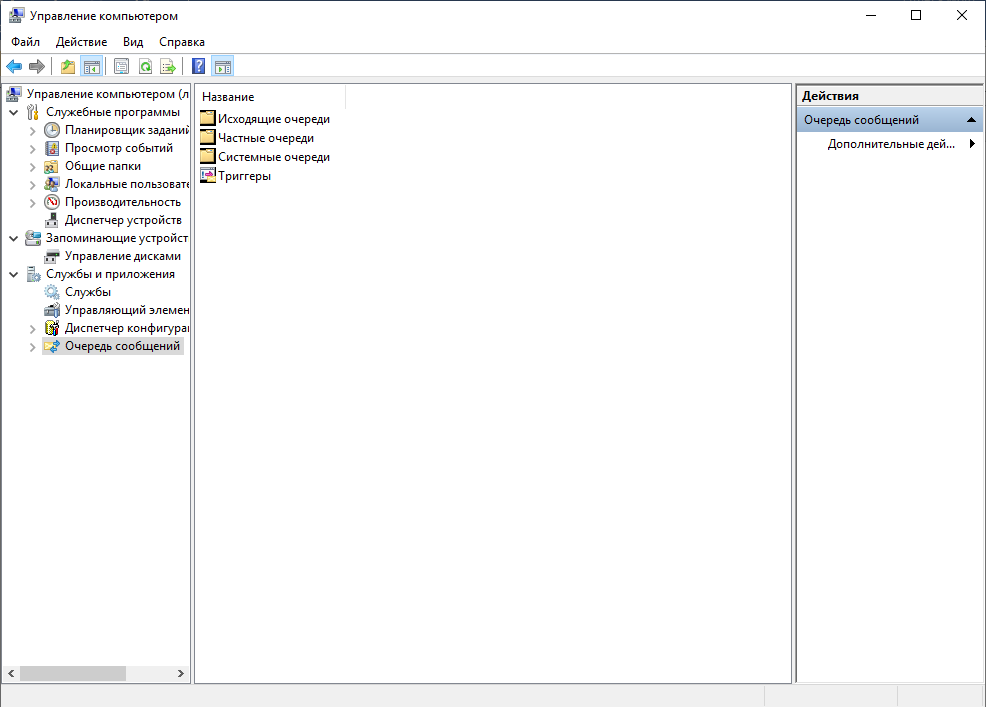
Установка очередей сообщений.

**Рисунок 1.1 Служба очередей сообщений**

# Задание 2

Для просмотра имеющихся на компьютере очередей сообщений в «Панели управления» перейдите в пункт «Администрирование», выберите ярлык «Управление компьютером». В открывшемся окне (в дереве навигации) щелкните по вершине «Службы и приложения» и выберите пункт «Очередь сообщений». Изучите какие очереди сообщений уже имеются на вашей машине.

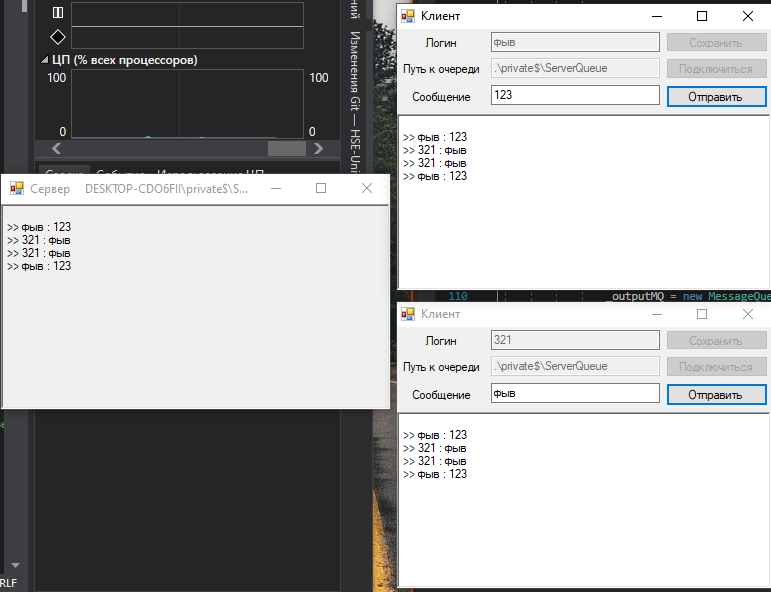
**Решение**



**Рисунок 2.1 Очереди сообщений на компьютере**

# Задание 3

Изучите разработанное приложение. Запустите исполняемый файл сервера и клиента на одной машине, затем перенесите один из компонентов распределенного приложения на другой вычислительный узел (лучше на ноутбук, т.к. администратор ЛВС вуза может запретить обмен сообщения между различными узлами ЛВС) и попробуйте его запустить. Объясните, почему приложение перестало работать? Запустите несколько клиентов на одной машине с сервером и попробуйте отправить сообщения серверу. В процессе изучения (тестирования) приложения просмотрите очереди сообщений компьютера и убедитесь, что сообщения действительно приходят в очередь.

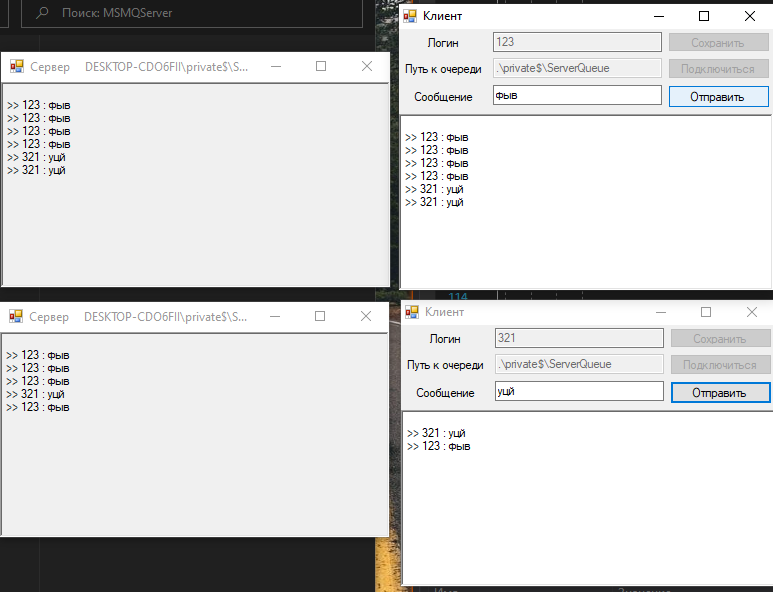
**Решение**

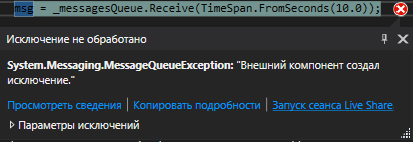
***Рисунок 3.1 Работа с приложением***

# Задание 4

Запустите несколько серверов на одной машине. Отправьте им сообщения от нескольких клиентов. Объясните, почему приложение перестало корректно работать.

**Решение**

Отправка сообщения отправляет по очереди сообщения на сервера. Но при простое программа вызывает исключение.

***Рисунок 4.1 Работа с приложением***

***Рисунок 4.2 Исключение***

# Задание 5

Модифицируйте приложение так, чтобы существовала возможность на сервере идентифицировать клиентов не по имени вычислительного узла, а по нику/логину пользователя.

**Решение**

if (string.IsNullOrEmpty(tbName.Text))

{

MessageBox.Show("Введите логин");

return;

}

else

{

\_clientName = tbName.Text;

btnConnect.Enabled = true;

if (MessageQueue.Exists(DIRECTORY\_NAME + \_clientName))

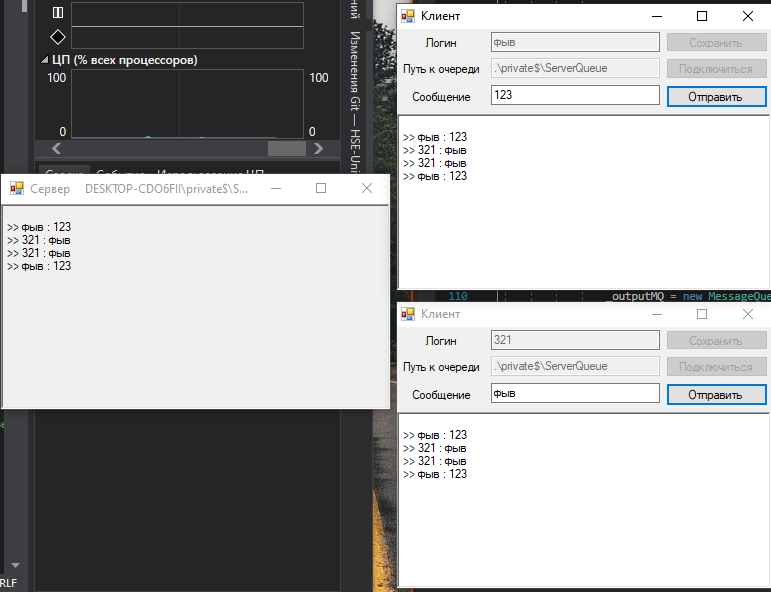
{

MessageBox.Show("Выберите другой логин");

return;

}

}



***Рисунок 5.1 Работа с приложением, логин***

# Задание 6

Модифицируйте приложение так, чтобы получился полноценный чат. Клиент может отправлять сообщения всем клиентам, участвующим в беседе. Для этого каждый клиент должен иметь возможность просмотра всех сообщений от всех клиентов, а сервер должен содержать список клиентов, которые хотят участвовать в беседе, чтобы каждый раз выполнять им рассылку сообщений.

Листинг сервера:

private void ReceiveMessage()

{

if (\_messagesQueue == null) return;

System.Messaging.Message msg = null;

bool write;

string messageResult;

MessageQueue outputMessageQueue;

// входим в бесконечный цикл работы с очередью сообщений

try

{

while (\_continue)

{

if (\_messagesQueue.Peek() != null) // если в очереди есть сообщение, выполняем его чтение, интервал до следующей попытки чтения равен 10 секундам

msg = \_messagesQueue.Receive(TimeSpan.FromSeconds(10.0));

if (!\_continue)

break;

write = true;

messageResult = (string)msg.Body;

//Добавить Клиента в список

if (messageResult.Contains("NEW\_MESSAGE\_QUEUE:"))

{

string name = messageResult.Split(':').ElementAt(1);

\_setClients.Add(name);

write = false;

}

//Удалить клиента из списка

if (messageResult.Contains("REMOVE\_MESSAGE\_QUEUE:"))

{

string name = messageResult.Split(':').ElementAt(1);

\_setClients.Remove(name);

write = false;

}

//Отправить сообщение всем

if (!write)

continue;

rtbMessages.Invoke((MethodInvoker)delegate

{

if (msg != null)

rtbMessages.Text += "\n >> " + msg.Body;// выводим полученное сообщение на форму

});

foreach (var x in \_setClients)

{

string path = NAME + x;

if (MessageQueue.Exists(path))

outputMessageQueue = new MessageQueue(path);

else

outputMessageQueue = MessageQueue.Create(path);

outputMessageQueue.Send(messageResult);

}

Thread.Sleep(500); // приостанавливаем работу потока перед тем, как приcтупить к обслуживанию очередного клиента

}

}

catch (ThreadInterruptedException ex)

{

MessageBox.Show(ex.Message);

}

}

Листинг клиента:

private void ReceiveMessage()

{

if (\_inputMQ == null)

return;

System.Messaging.Message msg = null;

try

{

while (\_continueReceivingMessages)

{

if (\_inputMQ.Peek() != null)

msg = \_inputMQ.Receive(TimeSpan.FromSeconds(10.0));

if (!\_continueReceivingMessages)

break;

string result = (string)msg.Body;

if (result == "REMOVE\_MESSAGE\_QUEUE")

continue;

rtbMessages.Invoke((MethodInvoker)delegate

{

if (msg != null)

rtbMessages.Text += "\n >> " + msg.Body;

});

Thread.Sleep(500);

}

}

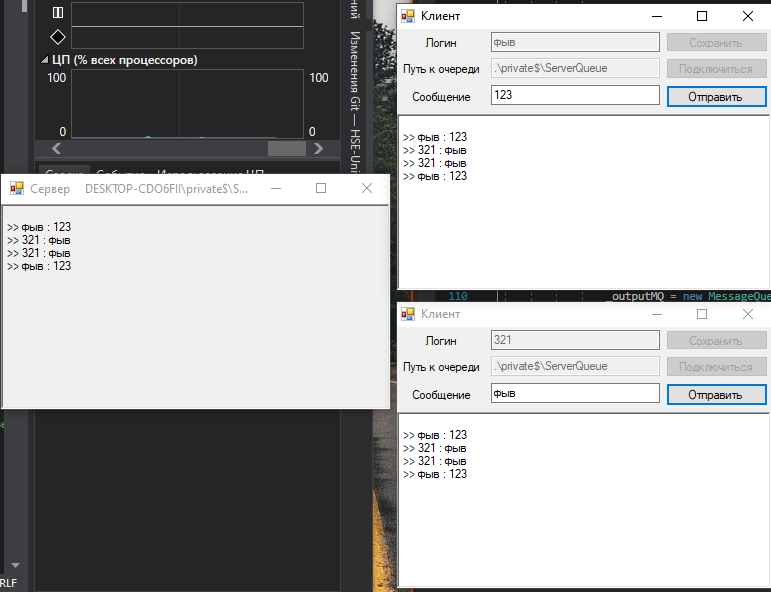
catch (ThreadInterruptedException ex)

{

MessageBox.Show(ex.Message);

}

}



***Рисунок 6.1 Работа чата***