Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий

институт

Кафедра «Информатика»

кафедра

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №2**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Преподаватель |  |  |  |  | М.О. Димухаметов |
|  |  |  | подпись, дата |  | инициалы, фамилия |
| Студент | КИ15–16Б, 031510065 |  |  |  | Т.В. Радионов |
|  | номер группы, зачетной книжки |  | подпись, дата |  | инициалы, фамилия |

Красноярск 2018

**ОГЛАВЛЕНИЕ**

[1 Постановка задачи 3](#_Toc530079536)

[2 Листинг 5](#_Toc530079537)

[3 Примеры работы программы 10](#_Toc530079538)

[4 Выводы 11](#_Toc530079539)

# Постановка задачи

*Бизнес-процесс B*

Этот бизнес-процесс принимает следующую информацию:

1. Имя сотрудника.
2. Company ID сотрудника.
3. Email адрес.
4. Текст письма.

Бизнес-процесс B отправляет запрос бизнес-процессу C. Отклик определяет:

1. Является ли адрес внешним.
2. Является ли отправитель родственником владельца.

После того, как бизнес-процесс B получает отклик, он определяет, нужно ли записать данные в BlackList.txt или SafeEmail.txt. Он выполняет такие действия:

* Если письмо отправлено на внешний адрес и отправитель НЕ родственник владельца, записать в BlackList.txt.
* Если письмо отправлено на внешний адрес и отправитель родственник владельца, записать в SafeEmail.txt.
* Если письмо отправлено на внутренний адрес, записать в SafeEmail.txt.

После этого бизнес-процесс отправляет запрос бизнес-операции E, чтобы записать информацию, которую он изначально получил, в соответствующий файл, основываясь на описанных условиях.

*Бизнес-процесс C*

Этот бизнес-процесс принимает следующую информацию:

1. Имя сотрудника.
2. ID сотрудника.
3. Email адрес.
4. Текст письма.

При отправке запросе в бизнес-операцию D бизнес-процесс C устанавливает:

1. Является ли адрес внешним.
2. Является ли сотрудник родственником владельца.

Бизнес-процесс C возвращает эти данные вызвавшему его компоненту.

(Не имплементируйте следующее требование во время упражнения. Вы выполните это вместе во время занятия.)

Также, он отправляет два блока данных, вместе с адресом, бизнес-процессу F, который выведет информацию в Address.txt. Address.txt отслеживает все адреса, на которые из компании были отправлены письма.

*Бизнес-процесс F*

Бизнес-процесс принимает следующую информацию и потом отправляет запрос компоненту E, чтобы добавить данные в файл Address.txt.

1. Email адрес.
2. Является ли адрес внешним.
3. Является ли отправитель родственником владельца.

# Листинг

Бизнес-процесс B

Class BProcess.DefineEmployeeSafety Extends Ens.BusinessProcess [ ClassType = persistent ]  
{  
  
Parameter ADAPTER = "EnsLib.File.OutboundAdapter";  
  
Property Adapter As EnsLib.File.OutboundAdapter;  
  
Method OnRequest(req As Msg.ReqLetter, Output res As Msg.ResWriteSuccess) As %Status  
{  
 #dim e As %Exception.StatusException  
 set res = ##class(Msg.ResWriteSuccess).%New()  
 try   
 {  
 set req1 = ##class(Msg.ReqLetter).%New()  
 set req1 = req  
 set res1 = ##class(Msg.ResEmployeeAddress).%New()  
 $$$ThrowOnError(..SendRequestSync("BProcess C", req1, .res1, 1, "CheckEmployeeAddress"))  
 if (res1.ExternalEmail = 1) & (res1.RelationshipToOwner = 0)  
 {  
 $$$ThrowOnError(..Adapter.PutLine("BlackList.txt", req.EmployeeName \_ ";" \_ req.EmployeeID \_ ";" \_ req.EmployeeEmail \_ $$$NL))  
 }  
 else  
 {  
 $$$ThrowOnError(..Adapter.PutLine("SafeEmail.txt", req.EmployeeName \_ ";" \_ req.EmployeeID \_ ";" \_ req.EmployeeEmail \_ $$$NL))  
 }  
 set req2 = ##class(Msg.ReqLetter).%New()  
 set req2 = req  
 $$$ThrowOnError(..SendRequestAsync("BOperation E", req2, 1, "AddLetter"))  
 }  
 catch e  
 {  
 return e.AsStatus()  
 }  
 return $$$OK  
}  
  
Method OnResponse(req As Ens.Request, ByRef res As Msg.ResWriteSuccess, callreq As Ens.Request, callres As Ens.Response, key As %String) As %Status  
{  
 if key = "AddLetter"  
 {  
 set res2 = ##class(Msg.ResWriteSuccess).%New()  
 set res2 = callres  
 set res.WriteSuccess = res2.WriteSuccess  
 }  
 else  
 {  
 return $$$ERROR($$$GeneralError, "key exception!")  
 }  
 return $$$OK  
}  
  
Storage Default  
{  
<Type>%Library.CacheStorage</Type>  
}

}  
  
Бизнес-процесс C

Class BProcess.CheckEmployeeAddress Extends Ens.BusinessProcess [ ClassType = persistent ]  
{  
  
Property RelationshipToOwner As %Integer [ InitialExpression = -1 ];  
  
Property ExternalEmail As %Integer [ InitialExpression = -1 ];  
  
Method OnRequest(req As Msg.ReqLetter, Output res As Msg.ResEmployeeAddress) As %Status  
{  
 #dim e As %Exception.StatusException  
 set res = ##class(Msg.ResEmployeeAddress).%New()  
 try   
 {  
 set req1 = ##class(Msg.ReqEmployeeID).%New()  
 set req1.ID = req.EmployeeID  
 $$$ThrowOnError(..SendRequestAsync("BOperation D", req1, 1, "CheckOnRelationshipToOwnerById"))  
 set req2 = ##class(Msg.ReqEmployeeEmail).%New()  
 set req2.Email = req.EmployeeEmail  
 $$$ThrowOnError(..SendRequestAsync("BOperation D", req2, 1, "CheckOnExternalEmail"))  
 }  
 catch e  
 {  
 return e.AsStatus()  
 }  
 return $$$OK  
}  
  
Method OnResponse(req As Ens.Request, ByRef res As Ens.Response, callreq As Ens.Request, callres As Ens.Response, key As %String) As %Status  
{  
 if key = "CheckOnRelationshipToOwnerById"  
 {  
 #dim res1 as Msg.ResEmployeeRelationshipToOwner = callres  
 set res.RelationshipToOwner = res1.RelationshipToOwner  
 set ..RelationshipToOwner = res1.RelationshipToOwner  
 //set res = callres  
 }  
 elseif key = "CheckOnExternalEmail"  
 {  
 #dim res2 as Msg.ResEmployeeExternalEmail = callres  
 set res.ExternalEmail = res2.ExternalEmail  
 set ..ExternalEmail = res2.ExternalEmail  
 //set res = callres  
 }  
 elseif key = "AddEmployeeAddress"  
 {  
 set res3 = callres  
 set ..RelationshipToOwner = -1  
 set ..ExternalEmail = -1  
 }  
 else  
 {  
 return $$$ERROR($$$GeneralError, "key exception!")  
 }  
 if (..RelationshipToOwner '= -1) && (..ExternalEmail '= -1)  
 {  
 $$$TRACE(..RelationshipToOwner \_ " " \_ ..ExternalEmail)  
 set req3 = ##class(Msg.ReqAddress).%New()  
 set req3.EmployeeEmail = req.EmployeeEmail  
 set req3.ExternalEmail = ..ExternalEmail  
 set req3.RelationshipToOwner = ..RelationshipToOwner  
 $$$ThrowOnError(..SendRequestAsync("BProcess F", req3, 1, "AddEmployeeAddress"))  
 }  
 return $$$OK  
}  
  
Storage Default  
{  
<Data name="CheckEmployeeAddressDefaultData">  
<Subscript>"CheckEmployeeAddress"</Subscript>  
<Value name="1">  
<Value>RelationshipToOwner</Value>  
</Value>  
<Value name="2">  
<Value>ExternalEmail</Value>  
</Value>  
</Data>  
<DefaultData>CheckEmployeeAddressDefaultData</DefaultData>  
<Type>%Library.CacheStorage</Type>  
}  
  
}

Бизнес-процесс F

Class BProcess.AddEmployeeAddress Extends Ens.BusinessProcess [ ClassType = persistent ]  
{  
  
Method OnRequest(req As Msg.ReqAddress, Output res As Msg.ResWriteSuccess) As %Status  
{  
 #dim e As %Exception.StatusException  
 set res = ##class(Msg.ResWriteSuccess).%New()  
 try   
 {  
 set req1 = ##class(Msg.ReqAddress).%New()  
 set req1.EmployeeEmail = req.EmployeeEmail  
 set req1.ExternalEmail = req.ExternalEmail  
 set req1.RelationshipToOwner = req.RelationshipToOwner  
 $$$ThrowOnError(..SendRequestAsync("BOperation E", req1, 1, "AddAddress"))  
 }  
 catch e  
 {  
 return e.AsStatus()  
 }  
 return $$$OK  
}  
  
Method OnResponse(req As Ens.Request, ByRef res As Msg.ResWriteSuccess, callreq As Ens.Request, callres As Ens.Response, key As %String) As %Status  
{  
 if key = "AddAddress"  
 {  
 set res1 = ##class(Msg.ResWriteSuccess).%New()  
 set res1 = callres  
 set res.WriteSuccess = res1.WriteSuccess  
 $$$TRACE(res.WriteSuccess)  
 }  
 else  
 {  
 return $$$ERROR($$$GeneralError, "key exception!")  
 }  
 return $$$OK  
}  
  
Storage Default  
{  
<Type>%Library.CacheStorage</Type>  
}  
  
}

# Примеры работы программы

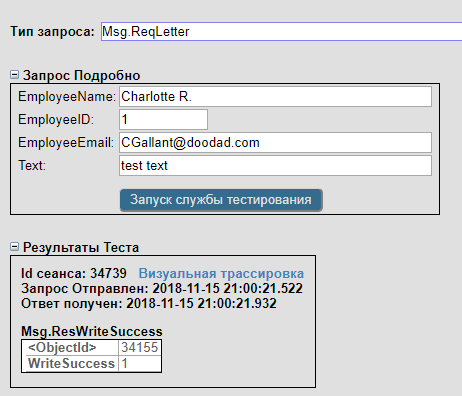


Рисунок 1 – Тестирование бизнес-процесса B

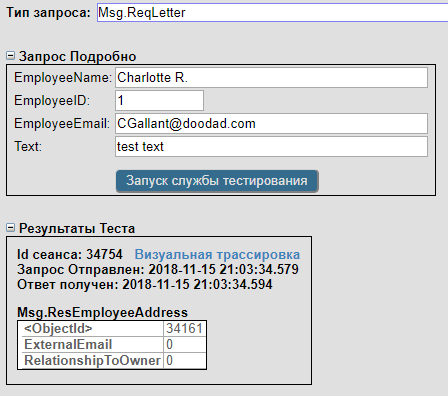


Рисунок 2 – Тестирование бизнес-процесса С

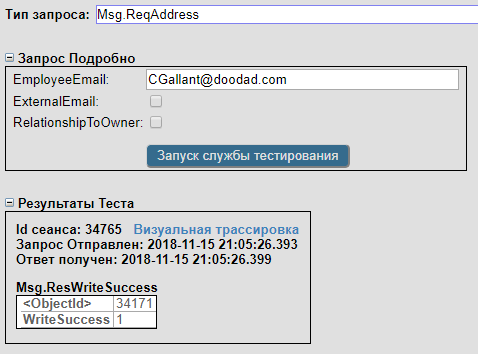


Рисунок 3 – Тестирование бизнес-процесса F

# Выводы

В ходе выполнения практической работы были получены навыки по созданию и управлению бизнес-процессами в Ensemble, а именно были реализованы вручную несколько процессов с использованием методов OnRequest и OnResponse, а также выполнены асинхронные и синхронные вызовы бизнес-процессов и бизнес-операции внутри бизнес-операции. Поставленная задача была выполнена в полном размере.