Team++

Requirements

And

Analysis

Compiled by:

Howard Grimberg

Pho Hale

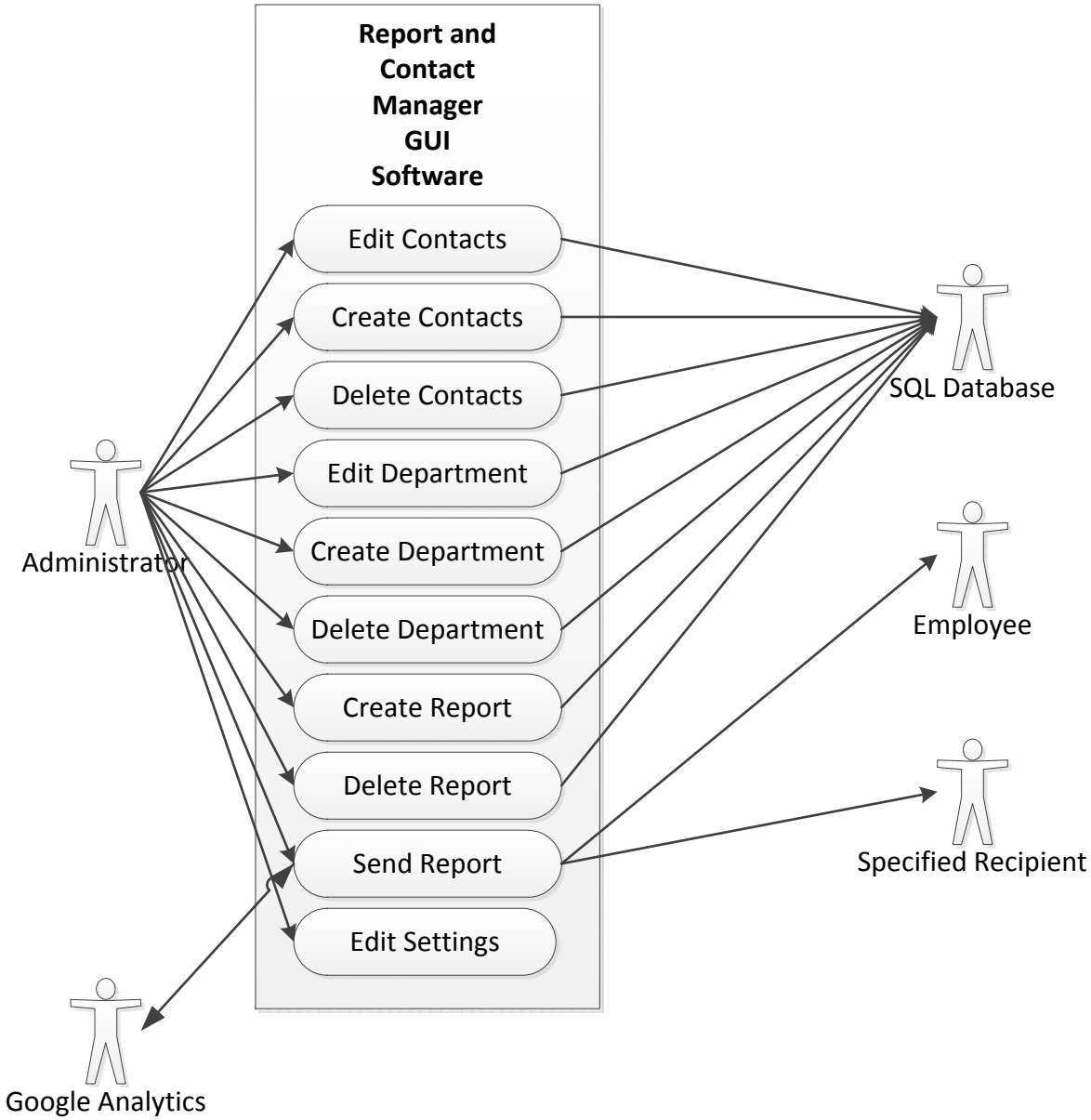
Tim Lamb

Allison Moses

Seth Polsley

Administrator	Main user of the GUI
Contact	Information within the system pertaining to an employee
Core Reporting API	A means of accessing and manipulating the data from Google Analytics for display
Dimensions	Breaks down metrics across some common criteria, such as date, city, browser, and operating system.
Employee	Person working within the company
GA Profile	Created for a website within a user account to track the website's metrics.
GA Query	A request of data from the profile selected in Google Analytics
Google Analytics (GA)	A utility provided by Google to analyze data gathered from a selected website profile.
GUI	Graphical User Interface
Max Results	The maximum number a returned selected dimension or metric can be.
Metrics	The individual measurements of visitor activity on your site. Options include visitors, number of new visits, and the amount of time a user spends on the site.
MySQL database	A collection of tables stored on a MySQL server
MySQL query	Command string that returns data from a MySQL database
MySQL server	A remote server used for hosting one or more databases
MySQL table	Comprised of columns and rows to hold data in a MySQL database.
Report	The collection of information returned from a query within the GUI.

Use-Case Diagram



Brief Description: The "Edit Contact" use case enables an administrator to change all or some of an existing contact information.

Step-by-step Description:

1. Administrator selects "Contacts" tab.
2. Administrator selects contacts to be changed from list of existing contacts.
3. Administrator left clicks the "Edit" button.
4. Administrator changes the fields that need to be updated.
5. Administrator left clicks the "Save" button to keep changes made.

Possible Alternative:
1. Database may fail to connect and program will display error.
2. Administrator may enter an invalid value into a field.
3. Administrator may changes values and attempt to exit without saving changes..
4. Administrator may change values but cancel all changes.
5. Administrator searches for contact's name via search bar instead of choosing from list.

Brief Description: The "Delete Contact" use case enables an administrator to delete all or some of the existing Contact(s).

Step-by-step Description:
1. Administrator selects "Contacts" tab.
2. Administrator selects contact to be deleted from list of existing Contacts.
3. Administrator left clicks the "Delete" button.
4. Window will appear asking the administrator "Are you sure you want delete (contact's name)?".
NOTE: this will delete contact from both contact list and department list and is irreversible!
5. Administrator left clicks the "Yes" button to keep changes made.

Possible Alternative:
1. Database can fail to connect and program will display error.
2. Administrator may enter an invalid value into a field.
3. Administrator may changes values and attempt to exit without saving changes..
4. Administrator may change values but cancel all changes.
5. Administrator may cancel deletion.
6. Administrator searches for contact’s name via search bar instead of choosing from list..

Brief Description: The "Create Contact" use case enables an administrator to add an Contact to the list of existing Contacts.

Step-by-step Description:
1. Administrator selects "Contacts" tab.
2. Administrator left clicks the "Create" button.
3. Administrator types in the contact information for the Contact being created into the appropriate fields.
4. Administrator left clicks "Save" to save new Contact

Possible Alternative:
1. Database can fail to connect and program will display error.
2. Administrator may fail to enter all necessary fields.
3. Administrator may changes values and attempt to exit without saving changes..
4.Administrator may change values but cancel all changes.

Brief Description: The "Edit Department" use case enables an administrator to change all or some of an existing departments' information.

Step-by-step Description:
1. Administrator selects "Departments" tab.
2. Administrator selects department to be changed from list of existing departments.
3. Administrator left clicks the "Edit" button.
4. If adding Contact to department, skip to Step 6..
5. If receiving Contact from department skip to Step 11.
6. Administrator changes the fields that need to be updated.
7. Administrator left clicks "Add Contact to Department" button.
8. Administrator selects Contact from list of existing Contacts.
9. Administrator left clicks "Ok" to return to department editor with Contact added to department list
10. Skip to Step 15 if administrator is done editing the department's information, continue to step 10 to remove contact from department, or return to Step 4 to keep editing.
11. Administrator left clicks "Remove Contact From Department" button
12. Administrator selects Contact from list of existing Contacts in the department.
13. Administrator left clicks "Ok" to return to department editor with Contacts removed from department list,or left clicks "Cancel" to return to department editor with Contacts still in the department.
14. If Administrator still needs to edit the department return to Step 4.
15. Administrator left clicks the "Save" button to keep changes made.

Possible Alternative:
1. Database can fail to connect and program will display error.
2. Administrator may fail to enter all necessary fields.
3. Administrator may changes values and attempt to exit without saving changes..
4. Administrator may change values but cancel all changes.
5. Administrator searches for Contact(s) by name via search bar.

Brief Description: The "Create Department" use case enables an administrator to add a

department to the list existing departments

Step-by-step Description:
1. Administrator selects "Departments" tab.
2. Administrator left clicks the "Create" button.
3. Administrator fills in department's information in correct fields.
4. Administrator left clicks "Add Contact" button.
5. Administrator selects Contacts from list of existing Contacts.
6. Administrator left clicks "Ok" button to add Contact to department's Contact list.
7. Repeat Steps 5 and 6 until all Contacts in department are added.
NOTE: Department must have at least one Contact to be saved.
8. Administrator left clicks "Save" button to create the department and return to list of existing departments.

Possible Alternative:
1. Database can fail to connect and program will display error.
2. Administrator may fail to enter all necessary fields.
3. Administrator may changes values and attempt to exit without saving changes..
4. Administrator may change values but cancel all changes.
5. Administrator may also search for Contact using name bar.

Brief Description: The "Delete Department" use case enables an administrator to delete department from the list of existing departments

Step-by-step Description:
1. Administrator selects "Departments" tab
2. Administrator selects department to be deleted
3. Administrator left clicks "Delete" button.
4. Administrator will be asked, "Are you sure you want to delete (department's name)?".
5. Administrator left clicks "Yes" button to delete department and returns to new list of existing departments.
NOTE: Contacts will be removed from department, but will not be deleted. The deletion of a department is irreversible.

Possible Alternative:
1. Database can fail to connect and program will display error.
2. Administrator may fail to enter all necessary fields.
3. Administrator may changes values and attempt to exit without saving changes.

4. Administrator may change values but cancel all changes.
5. Administrator may also search for Contact using name bar.

Brief Description: The "Create Report" use case enables an administrator to create a new report and save the report to Reports.

Step-by-step Description:
1. Administrator selects "Reports" tab
2. Administrator left clicks "Create" button.
3. Administrator inputs report information (start/end date, dimensions, metrics, sort by, filter, Title).
    NOTE: Report requires at least one metric.
4. Administrator clicks “Preview” to view the report before saving.
5. Administrator left clicks "Create report" button.
NOTE: Reports are saved in database.

6. Administrator left clicks "x" button to close report and is returned to list of reports in Reports folder.

Possible Alternative:
1. Database can fail to connect and program will display error.
2. Administrator may fail to enter all necessary fields.
3. Administrator may changes values and attempt to exit without saving changes..
4. Administrator may change values but cancel all changes.

Brief Description: The "Delete Report" use case enables an administrator to delete a report.

Step-by-step Description:
1. Administrator selects "Reports" tab
2. Administrator selects report to be deleted.
3. Administrator left Clicks "Delete Report" button.
4. Administrator is asked, "Are you sure you want to delete (report name)?"
5. Administrator left Clicks "Yes" button to delete a report and is returned to list of reports in Reports folder.

NOTE: There is no way to undelete a report.

Possible Alternative:
1. Database can fail to connect and program will display error.
2. Administrator may fail to enter all necessary fields.
3. Administrator may changes values and attempt to exit without saving changes..
4. Administrator may change values but cancel all changes.

Brief Description: The "Send Report" use case enables an administrator to send a report(s) to other Contacts or other specified recipients.

Step-by-step Description:
1. Administrator selects "Reports" tab.
2. Administrator selects report to be sent.
3. Administrator left clicks "Send" button.
4. If Administrator is sending to contact, Administrator left clicks "Add Contact to Send List" button.
5. Administrator selects Contact(s) to receive report.
6. Administrator left clicks "Add Contacts(s)" button to add contacts(s) to send list.
7. Repeat Steps 4,5, and 6 until all contacts intended to be sent to are added to send list.
8. If administrator wants to send report to an entire department, administrator left clicks add department.
9. Administrator selects department from list of departments.
10. Administrator left clicks "Add Email(s)" button to add email(s) to send list.
11. Administrator can click “Preview” to see the contents of the email/report before sending.
12. Repeat Steps 8,9, and 10 until all departments intended to be sent to are added to send list.
13. If administrator wants to send the report to specified recipient, administrator left clicks "Add Email Address" button.
14. Administrator fills in the "Email Address" field with the specified recipient's email address.
15. Administrator left clicks "Add Email" button to add email to send list.
16. Repeat Steps 13, 14, and 15 until as specified recipients intended to be sent to are added to send list.
17. Administrator left clicks "Send Report" button to send report to all people on send list.

NOTE: At least one contact, department, or specified recipient must be on send list to send report.

Possible Alternative:
1. Program may not be able to connect to mail server to send reports.
2. Database may fail to connect and will return an error.
3. Invalid recipient may be selected.
4. Administrator may get cancel the sending of a report.
5. Contact may also be found by using the name search function.
6. Department may also be found by using the department search function.

Brief Description: The “Edit Settings” use case enables the administrator to change or edit the SQL server, email server, and email account being used.

Step-by-Step Description:
1. Administrator selects the SQL database type via a combobox beside the “SQL Database Type:” label.
2. Administrator types the SQL Server address beside the label “SQL Server:”.
3 Administrator types the SQL Database’s name in the field beside “SQL Database Name:” label.
4. Administrator types his/her username for the SQL server in the field beside the “SQL Username:” label.
5. Administrator types his/her password for the SQL server in the field beside the “SQL Password:” label.
6. Administrator types the domain of his/her email client into the field beside the “Email Server:” label.
7. Administrator types his/her username for his/her email into the field beside the “Email Username:”.
8. Administrator types his/her password for his/her email into the field beside the “Email Password:” label.
9. Administrator clicks “Save Changes” button at the bottom of the GUI.
10. Administrator is prompt with a window that say, “Are sure you want to save changes to the settings?”
11. Administrator clicks “Yes, save”.
12. Administrator is returned to the “Settings” tab with the newly saved settings

Possible Alternative:
1. Administrator clicks “Revert Changes” and Administrator is returned Settings with last saved settings.
2. Administrator clicks “No, don’t save.” and Administrator is returned to the “Settings” tab with the settings he/she has not saved.

A. Add a Contact.
    a. Administrator selects "Contacts" tab.
    b. Administrator left clicks the "Create Contact" button.
    c. Administrator types in the contact information for the employee being created into the appropriate fields.

        Fields Include:

        - First Name:Jesse
        - Last Name:Pinkman
        - Telephone:1-505-654-3434
        - Email:jpinkman@vamanos.com
        - Birthday:05/15/1985

    d. Administrator left clicks "Save" to save new employee.
    e. A new record with the above information is created in the database.
    f. A dialog appears confirming the addition.

B. Edit an existing Contact.
    a. Administrator selects "Contact" tab.
    b. Administrator clicks in the search bar (next to the magnifying glass).
    c. Administrator enters “Cornish” (This is the last name).
    d. A list appears displaying all users with “Cornish” in any field (ex: name, phone number, email, etc.).
    e. Administrator selects the entry with the name “Conrad Cornish”.
    f. The right side of the window will be populated with “Conrad Cornish”.
    g. The Administrator left clicks the “Edit Contact” button.
    h. Administrator changes the telephone from “911” to “1-900-656-1224”.
    i. Administrator also changes the birthday from “01/01/1900” to “01/01/1970”.
    j. Administrator changes email from ccornish@miners.com to Cornishriotinto.com
    k. Administrator clicks save.
    l. A window appears explaining that the email field is incorrect.
    m. Administrator changes the email field to Cornish@riotinto.com.
    n. Administrator clicks save.
    o. A window appears confirming the save.

C. Remove an existing contact.
    a. Administrator selects "Contacts" tab.
    b. Administrator selects the search bar (next to the magnifying glass at the top).
    c. Administrator enters “Cornish” into the search bar.
    d. Administrator selects the entry with the name “Conrad Cornish”.
    e. Administrator the window beside the contact list is populated with “Conrad Cornish” ‘s contact information

f. Administrator clicks “Delete”.
g. A dialog appears to confirm the deletion.
h. Administrator clicks “Yes”.
i. The record is now deleted from the database.
j. All fields are now empty.

D. Add a new department.
   a. Administrator selects "Departments" tab.
   b. Administrator left clicks the "Create New Department" button.
   c. Administrator enters the following information.
      i. Department Name: Accounting
      ii. Department Telephone:1-888-555-5555
      iii. Department Email:acct@vamanos.com
   d. Administrator left clicks "Add Employee" button.
   e. Administrator searches for the name “Walter White” and selects the name returned.
   f. Administrator left clicks "Ok" button to add employee to department’s employee list.
   g. The list is now updated with the name “Walter White”.
   h. Administrator left clicks "Save" button to create the department and is returned to list of existing departments.

E. Edit an existing department.
   a. Administrator selects "Departments" tab.
   b. Administrator selects “Accounting” from the list of possible departments.
   c. The contact in formation for that department and employees in that department are populated.
   d. Administrator left clicks the "Edit Selected Department" button.
   e. Administrator left clicks "Add Employee to Department" button
   f. An employee search window appears. The name “Hank Schrader” is selected from the list.
   g. Administrator clicks “Add”.
   h. The name “Hank Schrader” now appears in the list.
   i. Administrator left clicks the "Save" button.
   j. The changes are saved to the database.
   k. Administrator is returned to the edit department screen.

F. Delete an existing department.
   a. Administrator selects "Departments" tab
   b. Administrator selects “Accounting” from the list of departments.
   c. The department contact information and list of employees in the department are retrieved from the database.
   d. Administrator left clicks "Delete" button.
   e. A dialog appears asking the user to confirm the deletion.
   f. Administrator left clicks "Yes" button to delete department and returns to new list of existing departments.

g. The department and its associations are deleted from the database.
h. Administrator is returned to the “Departments” screen.

G. Add a new report.
   a. Administrator selects "Reports" tab
   b. Administrator left clicks "Create" button.
   c. Administrator enters the following information:
      i. Start Date : 01/01/2012
      ii. End Date:06/01/2012
      iii. Dimensions: Operating System
      iv. Metric: Unique Visits
      v. The sort and filter fields are left empty.
   d. Administrator left clicks "Create report" button.
   e. Administrator is returned to the “Report” menu.

H. Delete an existing report.
   a. Administrator selects "Reports" tab
   b. Administrator left Clicks "Delete a Report" button.
   c. Administrator selects “Daily” report from list of reports.
   d. A dialog appears asking if the employee wants to delete the “Daily” report.
   e. Administrator left Clicks "Yes" button.
   f. The report is deleted.

I. Send a report.
   a. Administrator selects "Reports" tab.
   b. Administrator selects the “Weekly Report” report.
   c. Administrator left clicks "Send Report" button.
   d. Administrator clicks “Add employees to send”.
   e. A dialog appears with the list of employees.
   f. Administrator selects “John Doe” and clicks add.
   g. “John Doe” is now added to the receipts.
   h. Administrator clicks “Add departments to send”
   i. A dialog appears with the list of all possible departments.
   j. Administrator selects the “IT” department and clicks “Add”.
   k. Repeat Steps 4,5, and 6 till all employees intended to be sent to are added to send list.
   l. The list of receipts is updated to include the “IT” department.
   m. Administrator clicks “Send”.
   n. The associated report is generated and sent to all the associated email addresses.
   o. A dialog appears to confirm the successful send.

J. Edit report
   a. Administrator selects “Settings” Tab.
   b. Administrator selects “MySQL” from combobox beside “SQL Database Type:”.

c. Administrator types “hrreports05.ku.edu” into the text field beside “SQL Server:”.

d. Administrator types “HR_Reports” into the text field beside “SQL Database Name:”.

e. Administrator types “hramdin” into the text field beside “SQL Username:”.

f. Administrator types “password” into the text field beside “SQL Password:”.

g. Administrator types “mail.ku.edu” into the text field beside “Email Server:”.

i. Administrator types “suzyq” into the text field beside “Email Username:”.

j. Administrator types “ccr” into the text field beside “Email Password:”.

k. Administrator then left clicks “Save Changes”.

l. settings for the SQL server and email server are now saved saved.

Class: **MainWindowClass**
Responsibility: 1. Send message to **ReportGeneratorClass** to open.. 2. Send message to **ContactManagerClass** to open. 3. Generate a (unique)instanceUUID value. 4. Encapsulates all other classes. 5. Encapsulates all application level information.
Collaboration: 1. **ReportGeneratorClass**(Dependency) 2. **ContactManagerClass**(Dependency)

Class: **ReportGeneratorClass**
Responsibility: 1. Send message to **DatabaseConnectorClass** to start connection. 2. Send request for report definitions to **DatabaseConnectorClass**. 3. Receive report definitions from **DatabaseConnectorClass**. 4. Make a new **ReportClass** for each possible report definition. 5. Send message to **AnalyticsConnectorClass** class requesting for data. 6. Send message to **ContactManagerClass** to request data. 7. Send message to **MailManagerClass** to start. 8. Send message to **MailManagerClass** containing all info. for emails. 9. Process received data into a series of ready reports. 10. Binds received contacts to ready-made report(s). 11. Sends a request for contents of a department to **ContactManagerClass**. 12. Responsible for all report related functionality. 13. Save new **ReportClass** instances into database. 14. Respond to changes in related **ReportClass** Instances**.**
Collaboration: 1. **AnalyticsConnectorClass**(Dependency) 2. **DatabaseConnectorClass**(Dependency) 3. **MailerManagerClass** 4. **ReportClass** 5. **ContactManagerClass**

Class: **ContactManagerClass**

Responsibility:

1. Send message to **DatabaseConnectorClass** to open (if not opened).
2. Send queries to **DatabaseConnectorClass**.
3. Receive data from **DatabaseConnectorClass.**
4. Make a new **ContactClass** for each entry received from **DatabaseConnectorClass**.
5. Serialize all **ContactClass** instances and send to **DatabaseConnectorClass**.
6. Make a new **DepartmentClass** for each entry received from **DatabaseConnectorClass**.
7. Serialize all **DepartmentClass** instances and send to **DatabaseConnectorClass**.

Collaboration:

1. **DatabaseConnectorClass**(Dependency)
2. **ContactClass**
3. **DepartmentClass**

Class: **MailerManagerClass**

Responsibility:

1. Prepare all inbound information from **ReportGeneratorClass** into a series of **EmailMessageClass** instances.
2. Send message to **MailSenderClass** to start.
3. Push **EmailMessageClass** instances to **MailSenderClass.**

Collaboration:

1. **MailerSenderClass**
2. **EmailMessageClass**
3. **ReportGeneratorClass**

Class: **DatabaseConnectorClass**

Responsibility:

1. Load **DatabaseSettingsClass**.
2. Connect to database management system.
3. Handles are database related traffic.
4. Translate queries from **ReportGeneratorClass** into database queries.
5. Translate queries from **ContactManagerClass** into database queries.

Collaboration:

1. **DatabaseSettingsClass**(Dependency)
2. **ReportGeneratorClass**
3. **ContactManagerClass**

Class: **DepartmentClass**

Responsibility:

1. Hold data for department-related information.
2. Respond to requests to various pieces of itself from **ContactManagerClass**.
3. Respond to changes to itself as requested by **ContactManagerClass**.
4. Notify **ContactManagerClass** when changes are successful and can be written to database.
5. Interacts with **DepartmentAssociationClass** to determine contact-department associations.

Collaboration:

1. **DepartmentAssociationClass**(Associative Class)
2. **ContactManagerClass**

Class: **EmailMessageClass**

Responsibility:

1. Encapsulate all data related to each email message.
2. Receive change requests from **MailManagerClass**.
3. Send message to **MailManagerClass** when it is ready to be sent.

Collaboration:

1. **MailManagerClass**

Class: **AnalyticsConnectorClass**

Responsibility:

1. Send requests for data to **Google Analytics**.
2. Process incoming data from **Google Analytics.**
3. Receive requests from **ReportGeneratorClass** and translate into requests ready to be sent to **Google Analytics**.
4. Send received to **ReportGeneratorClass**.
5. Notify **ReportGeneratorClass** when data is ready to be retrieved.

Collaboration:

1. **ReportGeneratorClass**
2. **Google Analytics** (external)

Class: **MailSenderClass**

Responsibility:

1. Receive ready **EmailMessageClass** instances from **MailManagerClass**.
2. Load settings from **MailSenderSettingsClass**.
3. Send emails by reading each **EmailMessageClass** instance.

Collaboration:

1. **EmailMessageClass**
2. **MailSenderSettingsClass**
3. **MailManagerClass**

Class: **ContactClass**

Responsibility:

1. Respond to requests for information from **ContactManagerClass.**
2. Process inbound changes as requested by **ContactManagerClass**.
3. Notify **ContactManagerClass** when it is safe to write changes to database.
4. Notify **ContactManagerClass** when data for the requested information does not exists.
5. References **DepartmentAssociationClass** to determine contact-department associations.

Collaboration:

1. **DepartmentAssociationClass**
2. **ContactManagerClass**

Class: **ReportClass**

Responsibility:

1. Respond to changes in the report definition from **ReportGeneratorClass**.
2. Notify **ReportGeneratorClass** when it is safe to write changes back to database.
3. Return necessary parameters for a given report as request by **ReportGeneratorClass**.

Collaboration:

1. **ReportGeneratorClass**

Class: **DepartmentAssociationClass**

Responsibility:

1. Associate relationships between **DepartmentClass** and **ContactClass**
2. Respond to change requests from either **DepartmentClass** or **ContactClass**.
3. Return employeeId/departmentId to either **ContactClass** or **DepartmentClass**.

Collaboration:

1. **DepartmentClass**
2. **ContactClass**

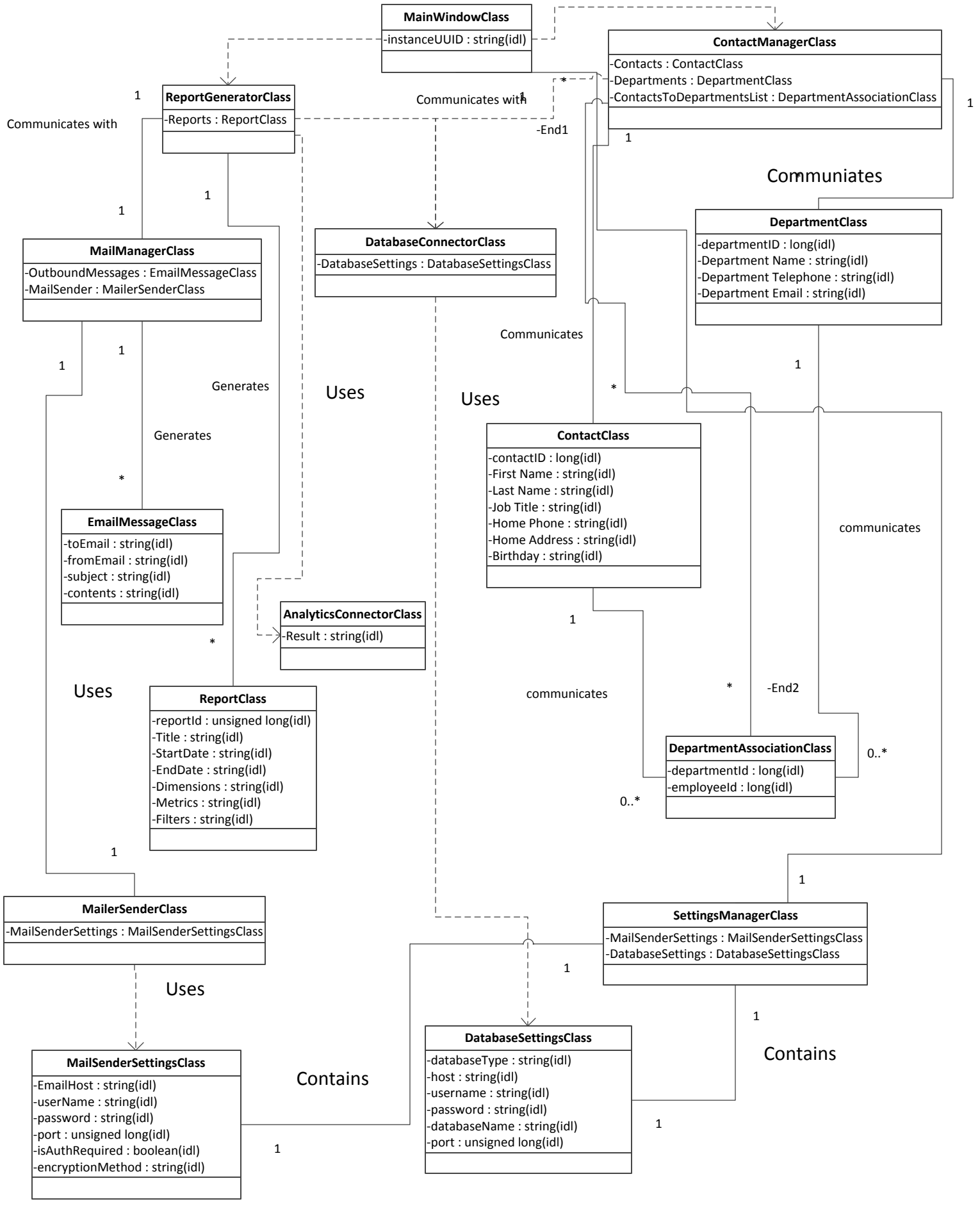
Class: **MailSenderSettingsClass**

Responsibility:

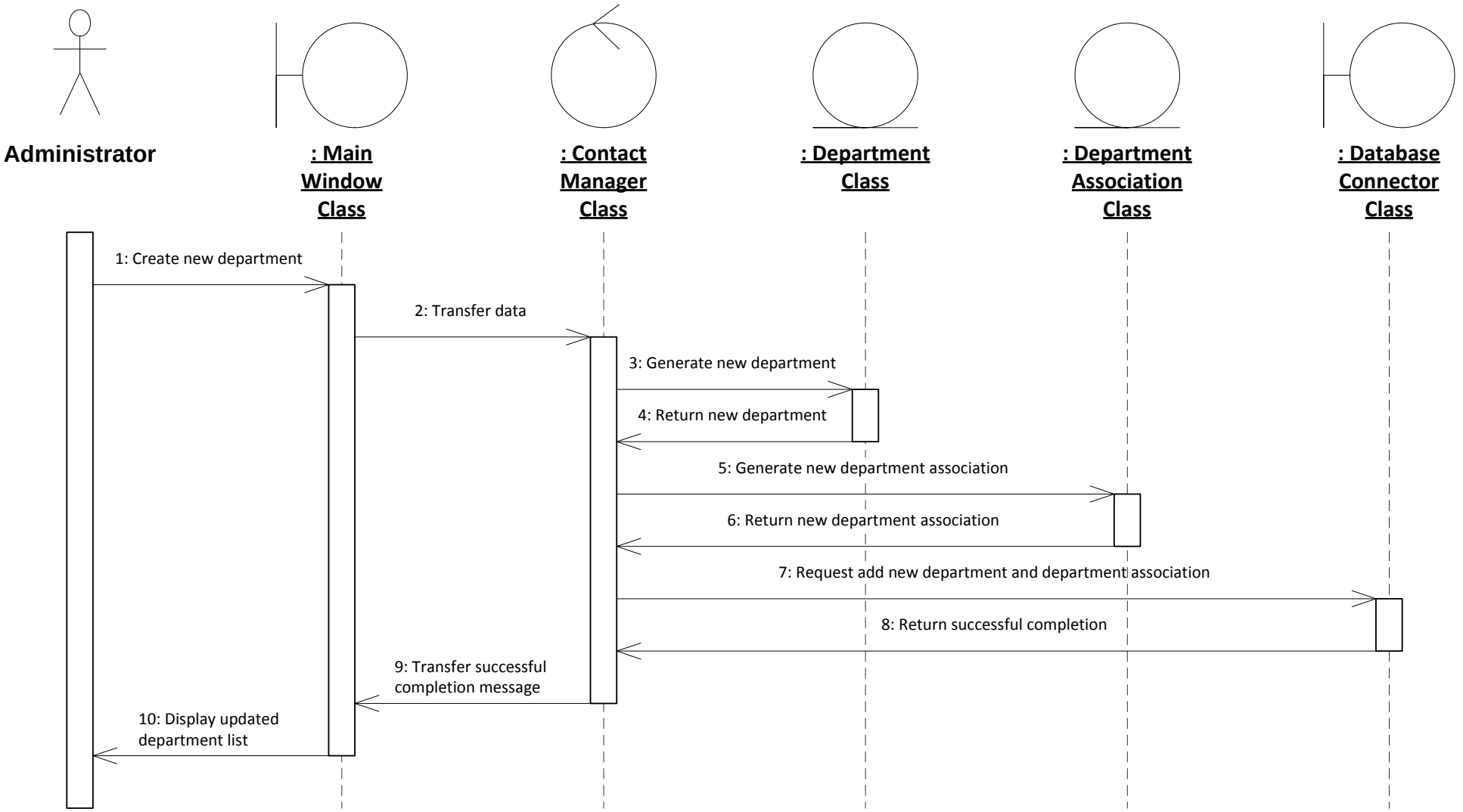
1. Respond to inbound changes from **MailSenderClass** to email related settings.
2. Return email related settings to **MailSenderClass**.

Collaboration:

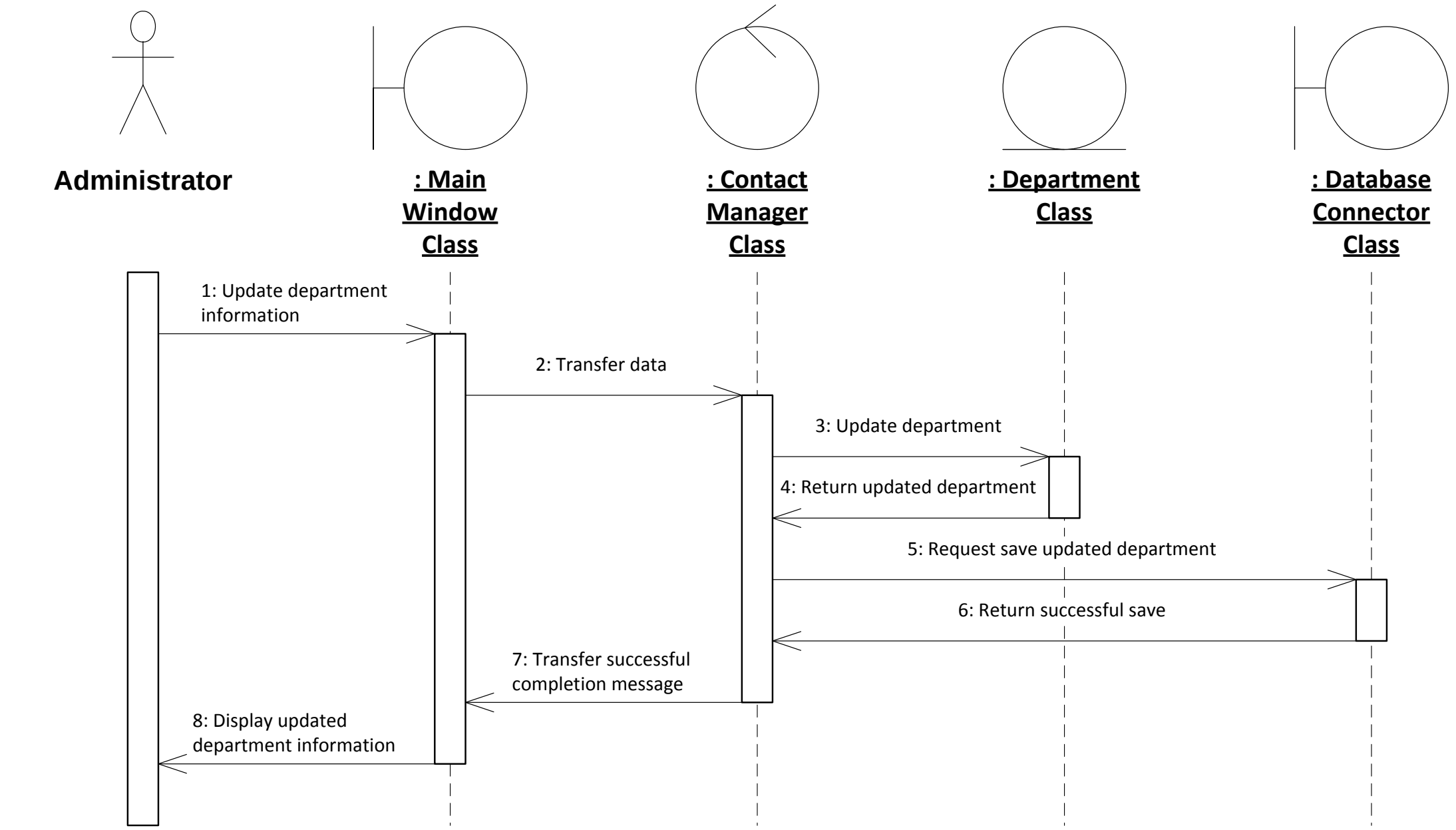
1. **MailSenderClass**


MainWindowClass
-instanceUUID : string(idl)
ContactManagerClass
-Contacts : ContactClass
-Departments : DepartmentClass
-ContactsToDepartmentsList : DepartmentAssociationClass
ReportGeneratorClass
-Reports : ReportClass
Communicates with
Communicates with
-End1
Communiates
MailManagerClass
-OutboundMessages : EmailMessageClass
-MailSender : MailerSenderClass
DatabaseConnectorClass
-DatabaseSettings : DatabaseSettingsClass
DepartmentClass
-departmentID : long(idl)
-Department Name : string(idl)
-Department Telephone : string(idl)
-Department Email : string(idl)
Communicates
Generates
Generates
Uses
Uses
ContactClass
-contactID : long(idl)
-First Name : string(idl)
-Last Name : string(idl)
-Job Title : string(idl)
-Home Phone : string(idl)
-Home Address : string(idl)
-Birthday : string(idl)
EmailMessageClass
-toEmail : string(idl)
-fromEmail : string(idl)
-subject : string(idl)
-contents : string(idl)
AnalyticsConnectorClass
-Result : string(idl)
communicates
Uses
ReportClass
-reportId : unsigned long(idl)
-Title : string(idl)
-StartDate : string(idl)
-EndDate : string(idl)
-Dimensions : string(idl)
-Metrics : string(idl)
-Filters : string(idl)
communicates
-End2
DepartmentAssociationClass
-departmentId : long(idl)
-employeeId : long(idl)
MailerSenderClass
-MailSenderSettings : MailSenderSettingsClass
SettingsManagerClass
-MailSenderSettings : MailSenderSettingsClass
-DatabaseSettings : DatabaseSettingsClass
Uses
Contains
Contains
MailSenderSettingsClass
-EmailHost : string(idl)
-userName : string(idl)
-password : string(idl)
-port : unsigned long(idl)
-isAuthRequired : boolean(idl)
-encryptionMethod : string(idl)
DatabaseSettingsClass
-databaseType : string(idl)
-host : string(idl)
-username : string(idl)
-password : string(idl)
-databaseName : string(idl)
-port : unsigned long(idl)

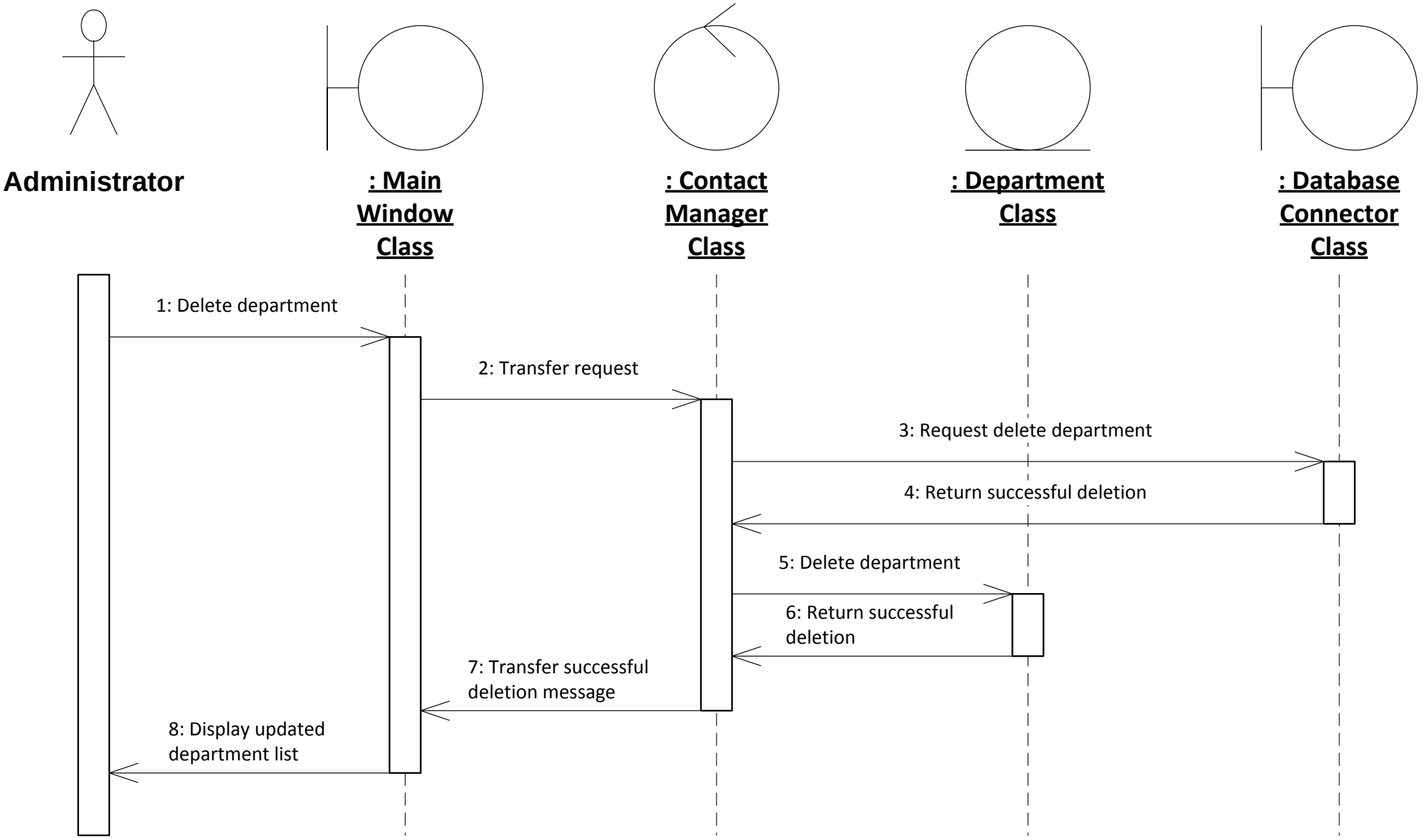
CreateDepartment



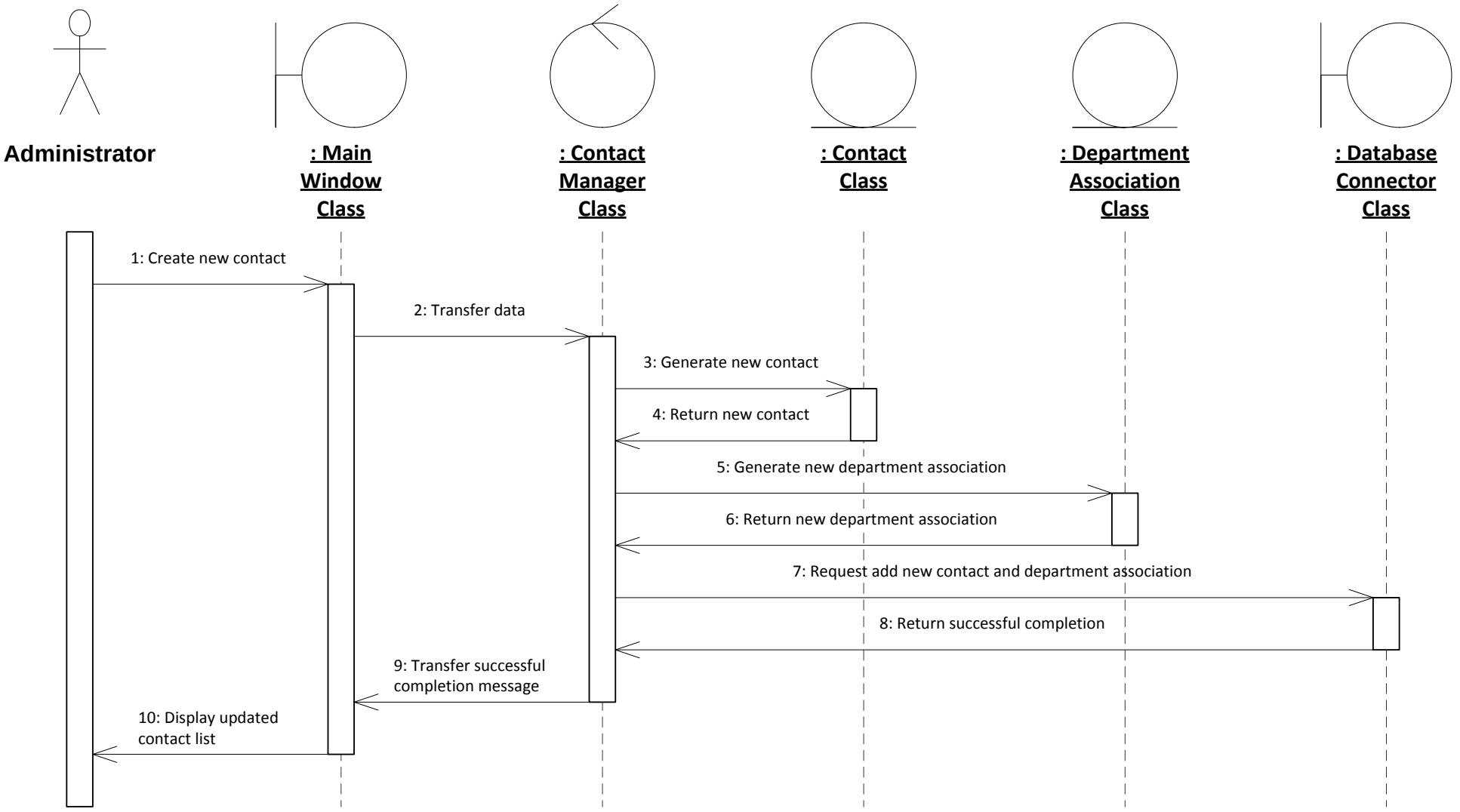
EditDepartment



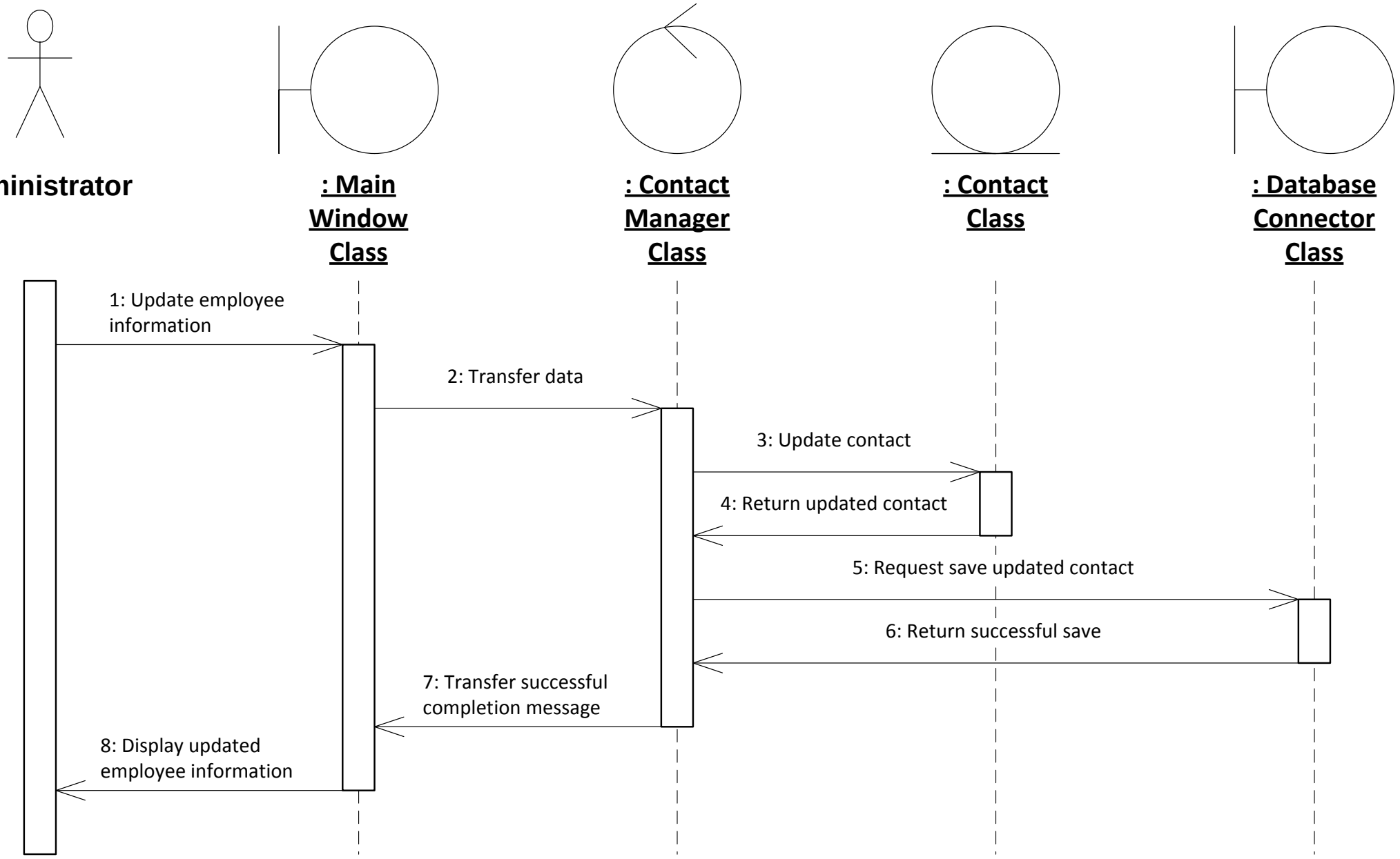
DeleteDepartment



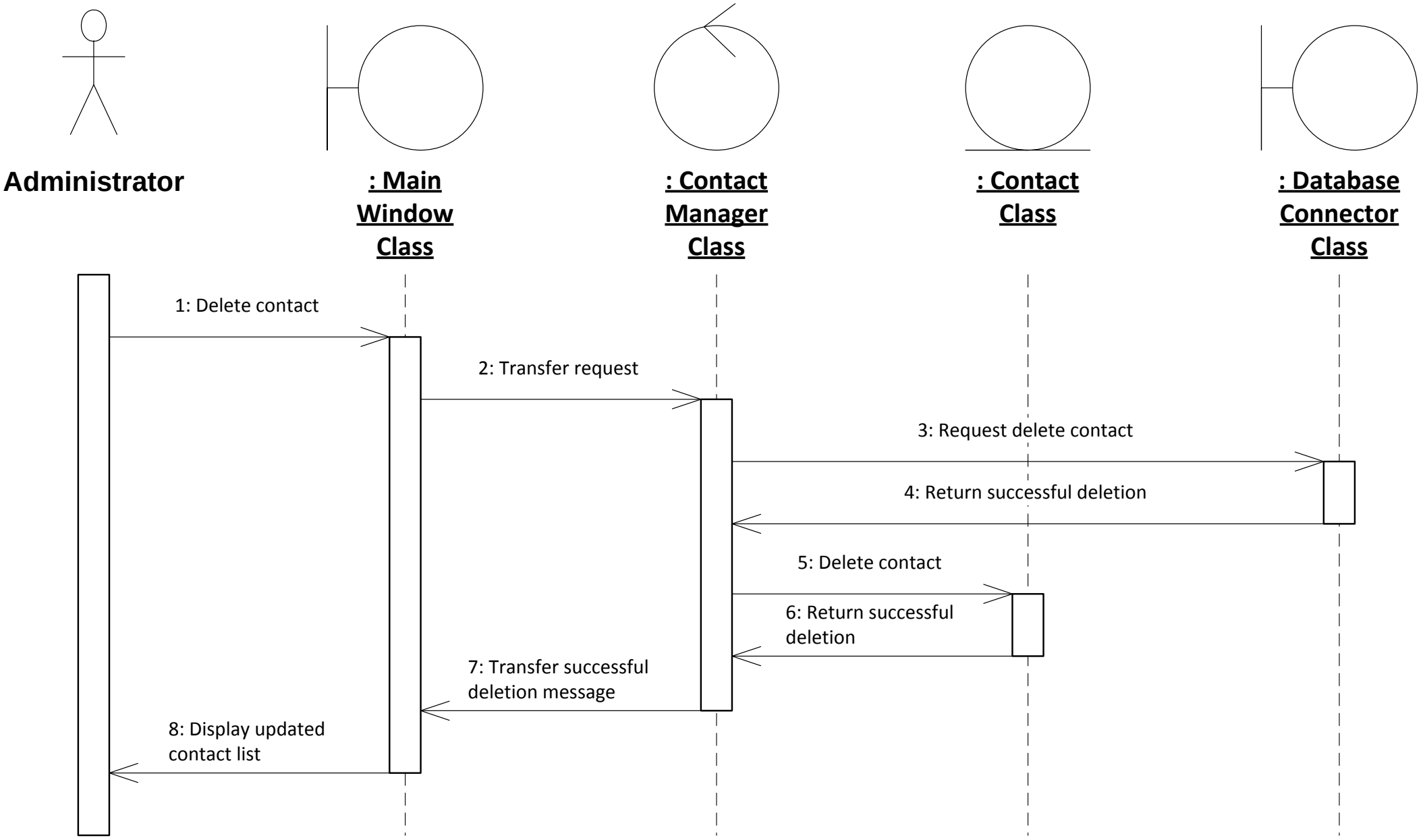
CreateEmployee



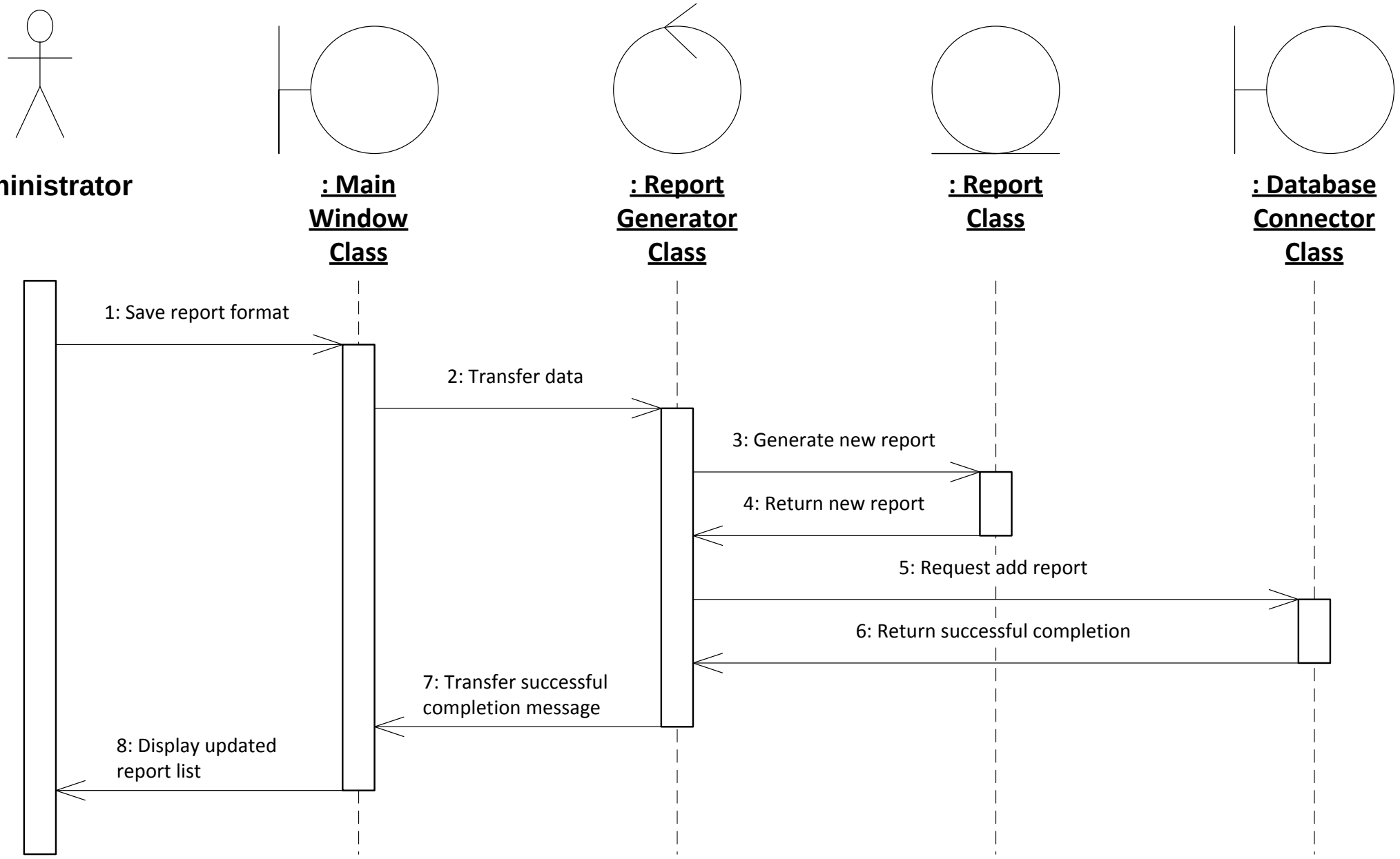
EditEmployee



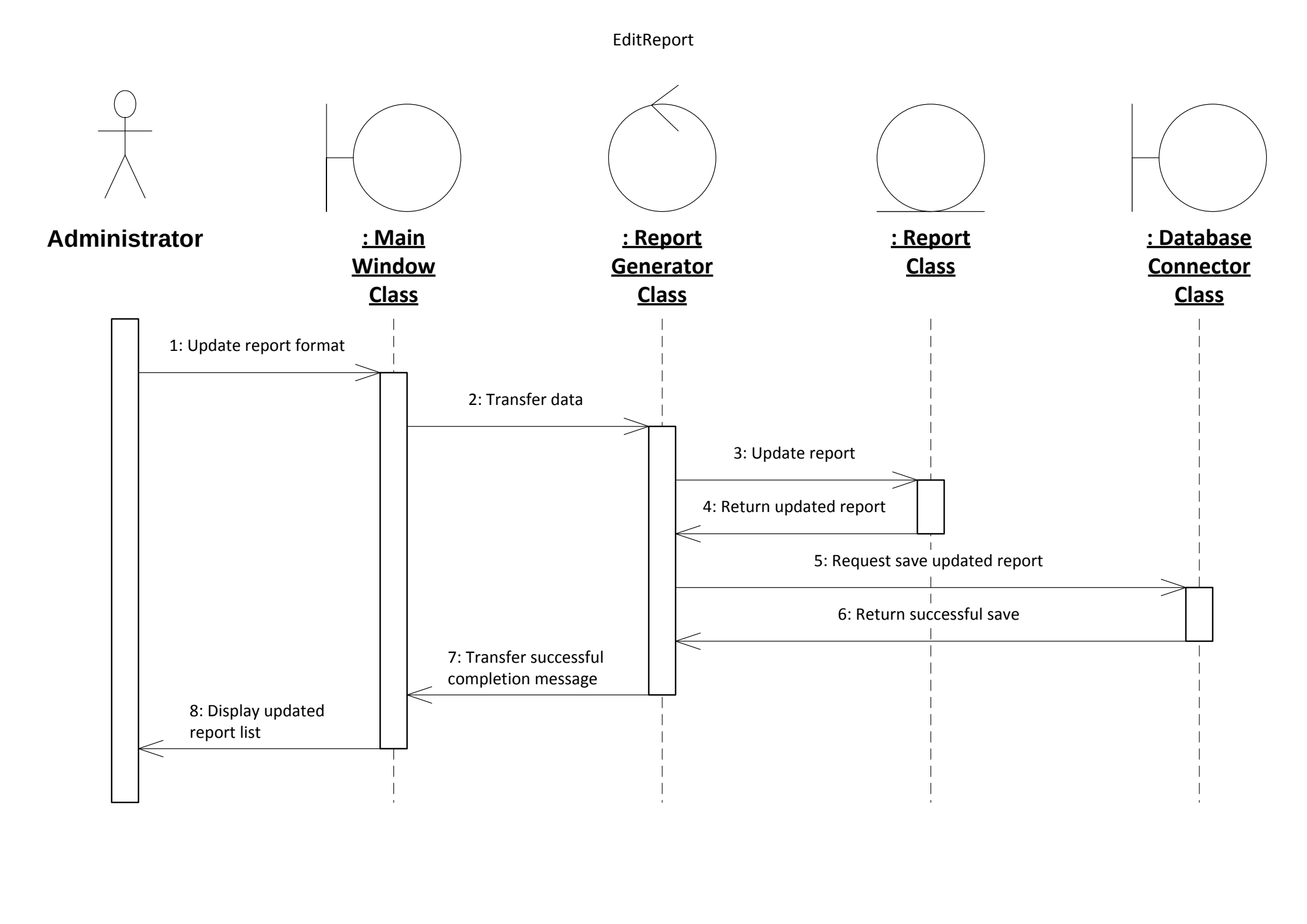
DeleteEmployee



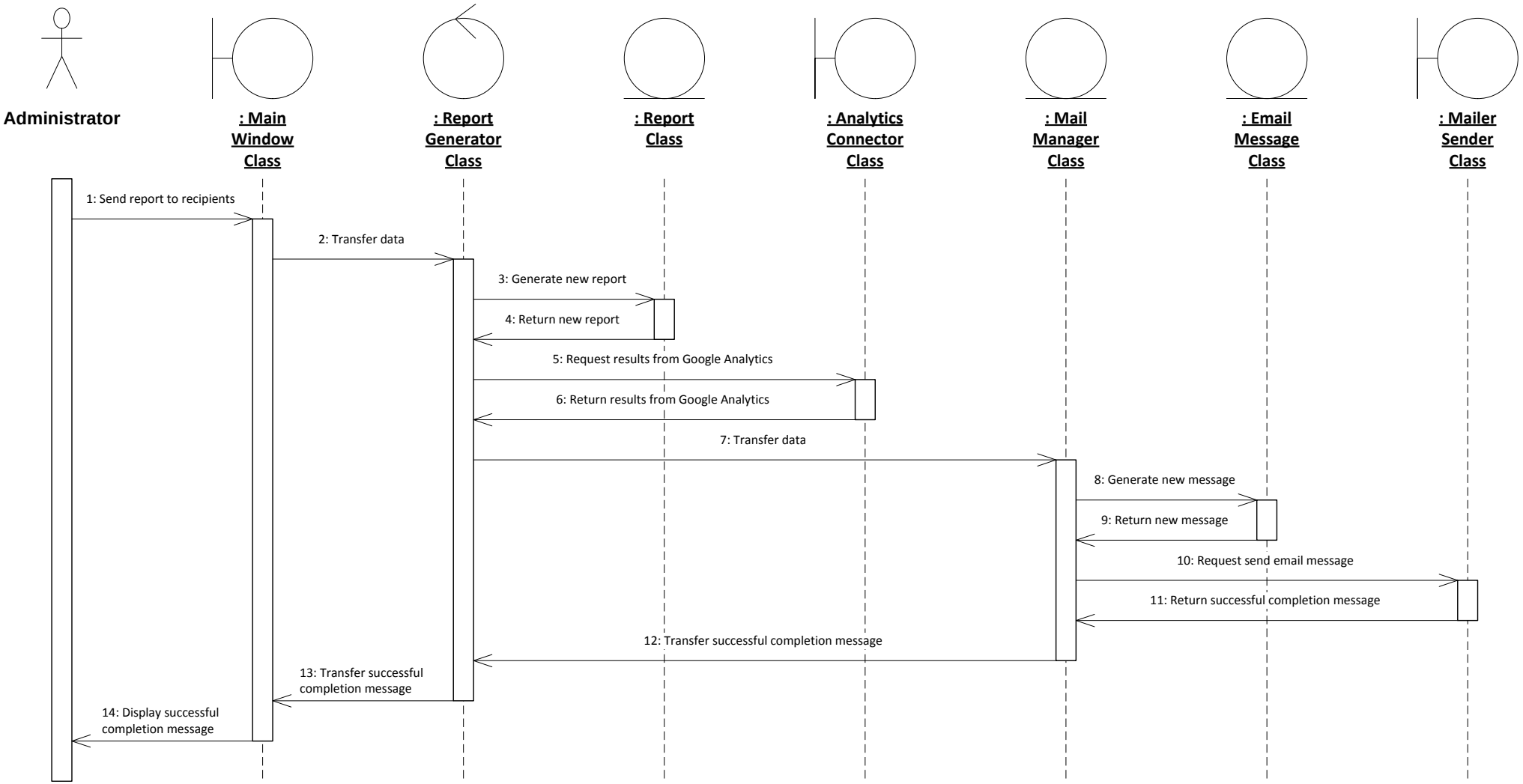
CreateReport



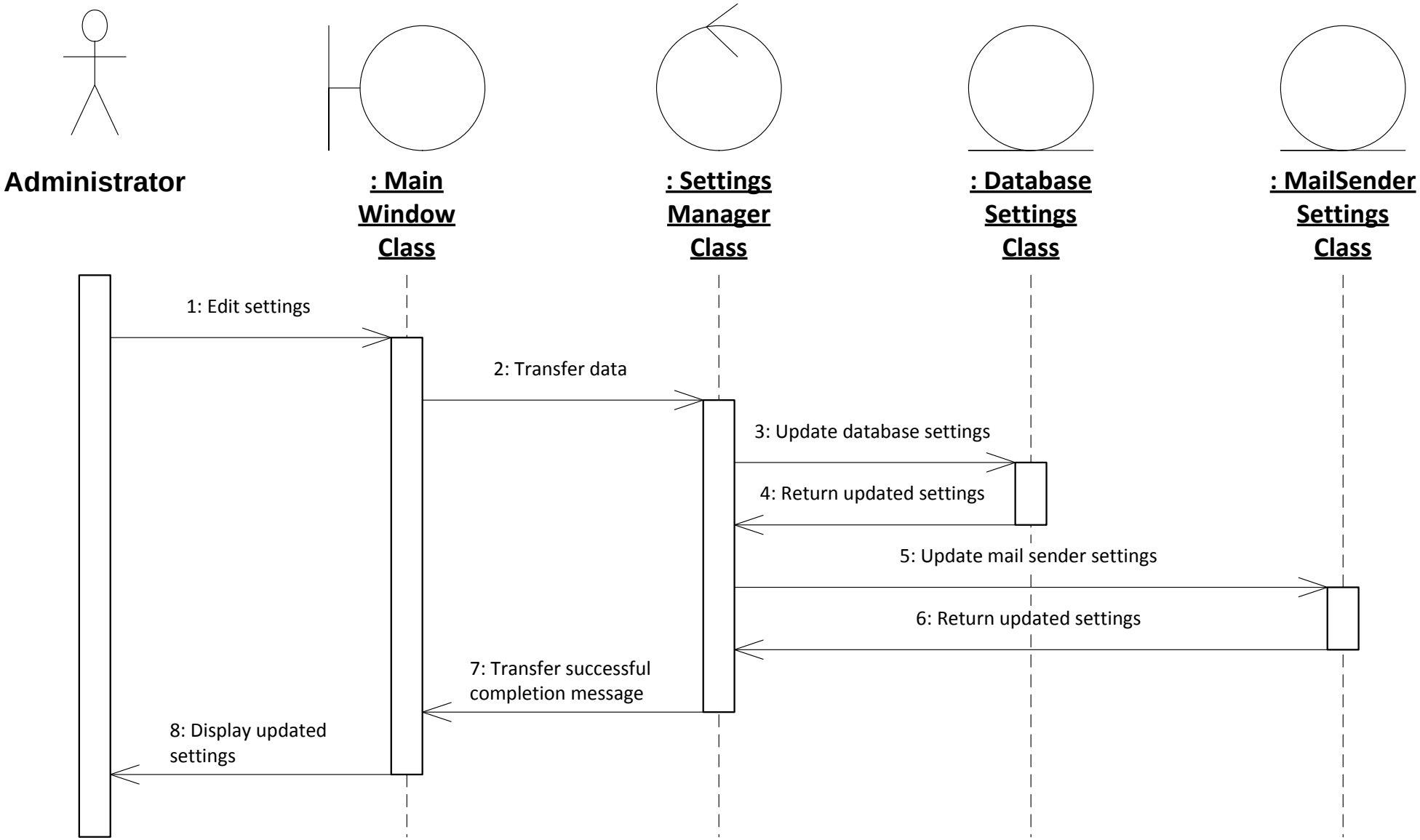
EditReport


Administrator
: Main Window Class
: Report Generator Class
: Report Class
: Database Connector Class
1: Update report format
2: Transfer data
3: Update report
4: Return updated report
5: Request save updated report
6: Return successful save
7: Transfer successful completion message
8: Display updated report list

SendReport


Administrator
: Main Window Class
: Report Generator Class
: Report Class
: Analytics Connector Class
: Mail Manager Class
: Email Message Class
: Mailer Sender Class
1: Send report to recipients
2: Transfer data
3: Generate new report
4: Return new report
5: Request results from Google Analytics
6: Return results from Google Analytics
7: Transfer data
8: Generate new message
9: Return new message
10: Request send email message
11: Return successful completion message
12: Transfer successful completion message
13: Transfer successful completion message
14: Display successful completion message

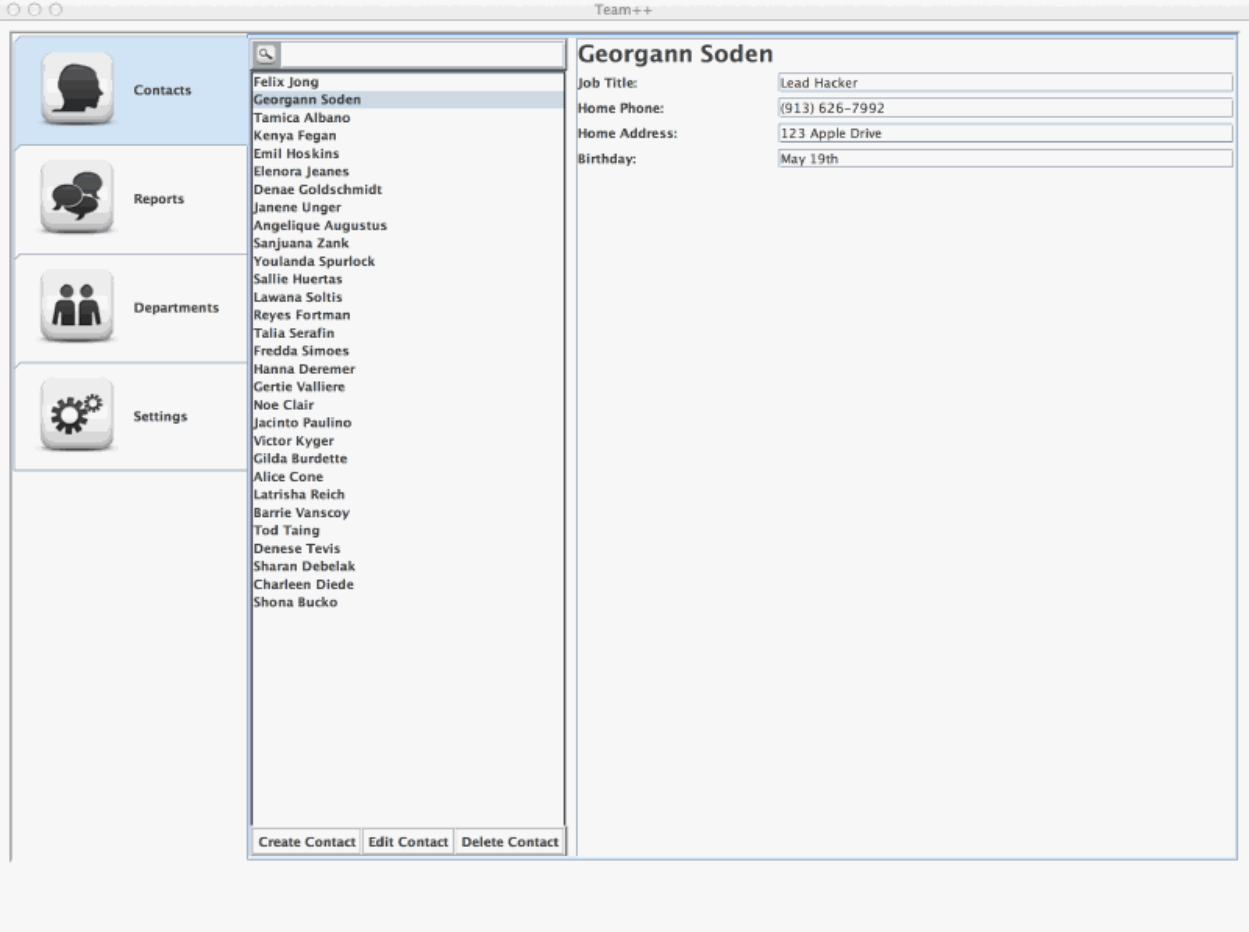
EditSettings



GUI Descriptions:

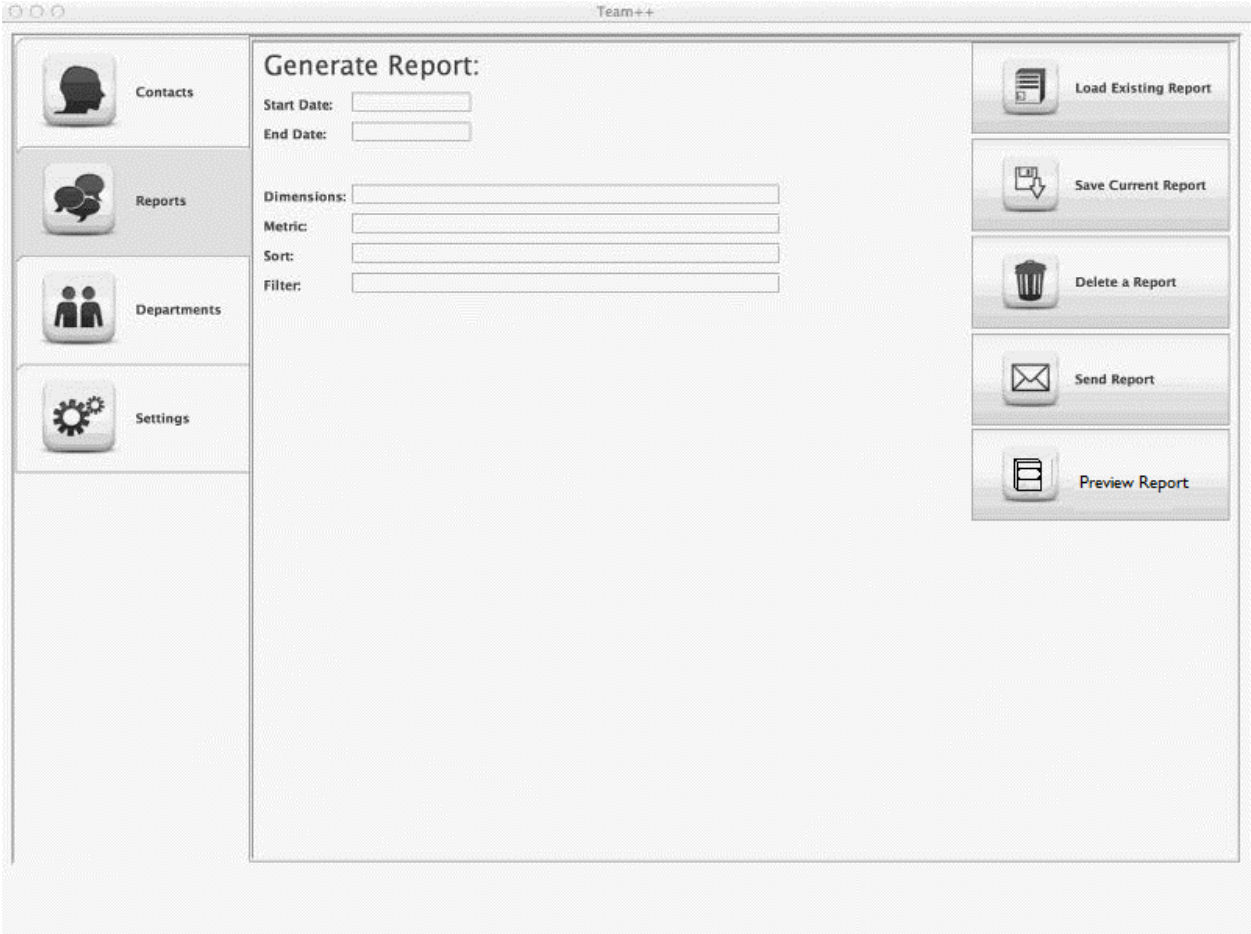
The main interface consists of four tabs, denoted both visually with an icon and with a textual label.

Contacts Tab:



Under the Contacts tab, the user is presented with a list of users in the left pane, and the selected user's information in right. On the top of the left contacts pane is a text field which allows the user to search for an existing contact. This bar filters the users below in real time, narrowing the results dynamically as they are typed.
When a user's name is selected from the list, the user's information is immediately displayed in the rightmost pane. At the bottom of the left contacts pane rests a toolbar containing a button for creating a new contact. When this button is clicked by the user, the pane to the right is populated with editable fields, which can be used to input information about the employee.

Reports Tab:



Under the Reports tab the user is presented with a single panel. On the left-hand side of this panel exists the user editable text fields. One of these exists for each parameter: From top to bottom:

- Start Date
- End Date
- Dimensions
- Metric
- Sort
- Filter

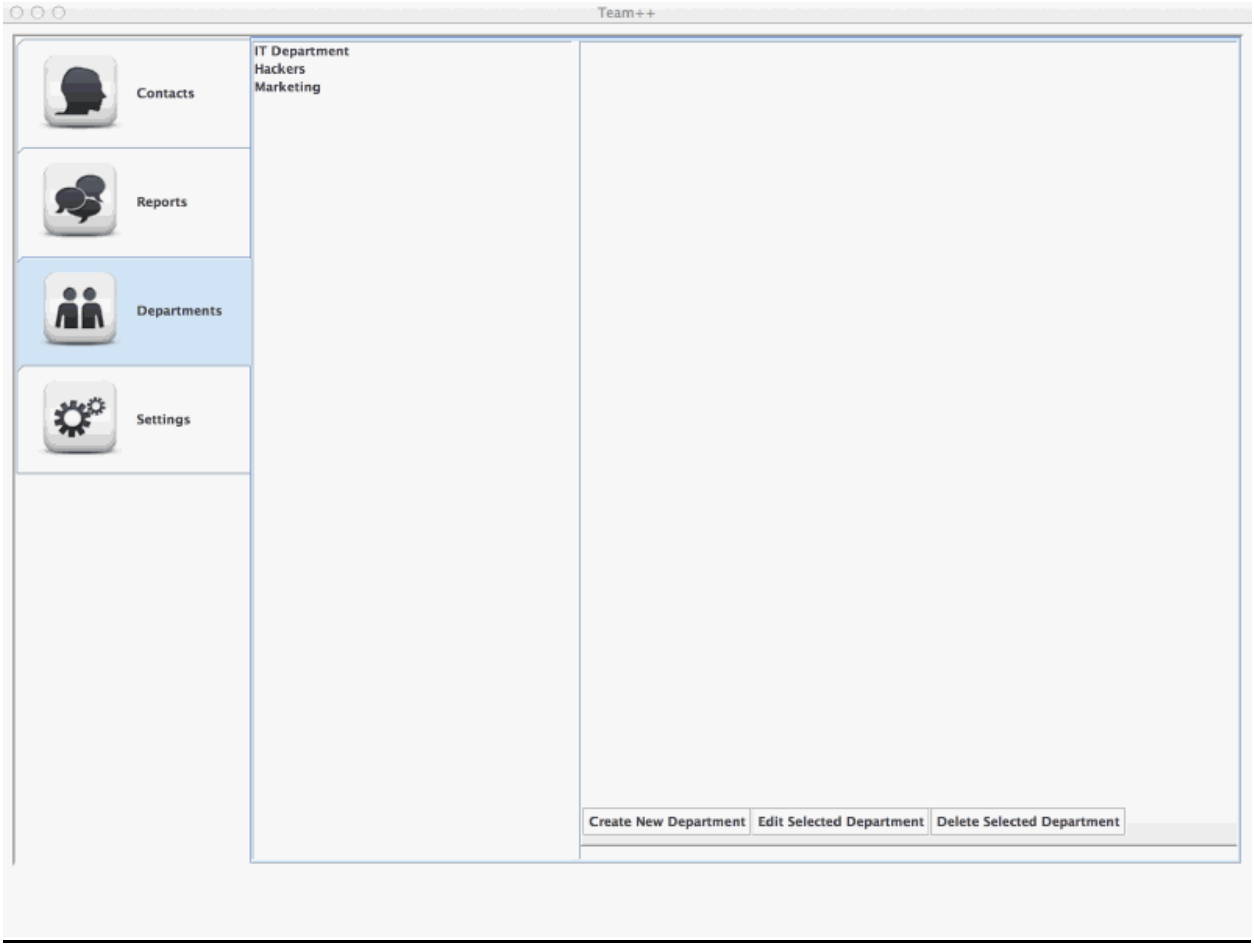
On the right-hand side of this panel the user will find four buttons. From top to bottom, these are:

- Load Existing Report: This opens up a popup panel that contains a list of existing reports saved on the SQL server. When a report is chosen, the leftmost information fields will be populated with its contained data.
- Save Current Report: This opens up a popup panel that again contains a list of existing reports saved on the SQL server. Now present is a text field in which a

name for a new report can be added, and the user's fields will be stored to the server under the specified name.

- Delete a Report: This option opens up the same popup panel that contains a list of existing reports saved on the SQL server. This time a delete button is present allowing the user to delete a single or multiple selected reports from the server.
- Send Report: This button opens up a mail client like interface, allowing the user to select a previously saved report and specify recipients.

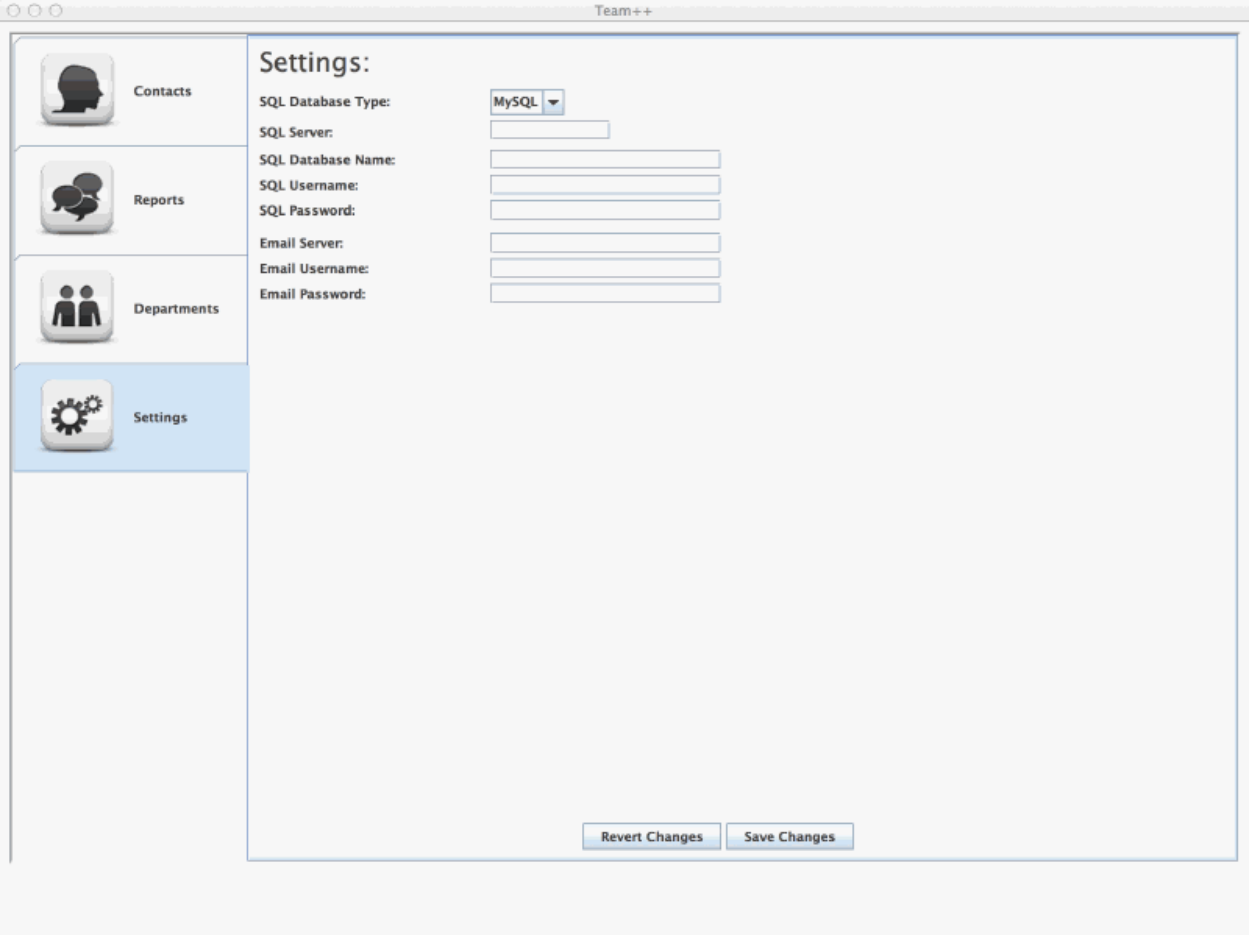
Departments Tab:



Under the Departments tab we find an interface very similar to the one under the contacts tab. It is again split into two panes, with the departments listed on the left and the department information listed on the right. The department information also includes the list of employees in the department. On the rightmost pane, there is a toolbar at the bottom containing three buttons. From left to right these are:

- Create New Department
- Edit Selected Department
- Delete Selected Department.

Settings Tab:



Finally, is the settings tab.

This tab contains user-editable text fields for email and SQL. It allows the user to dynamically edit SQL database type, name, and authentication credentials as well as SMTP server name and credentials. At the bottom are "Revert Changes" and "Save Changes" buttons.