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**

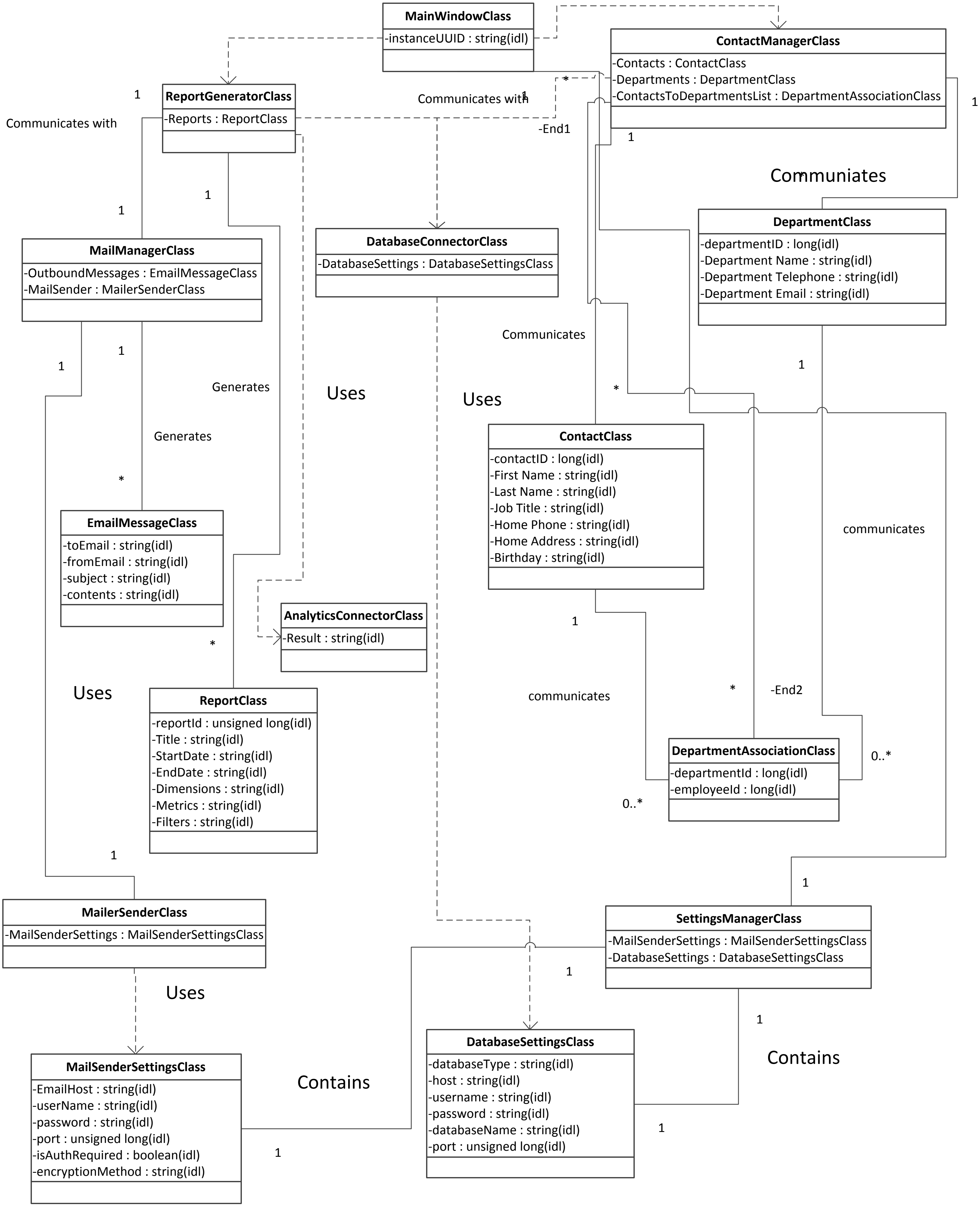
Class: **MailSenderSettingsClass**

Responsibility:

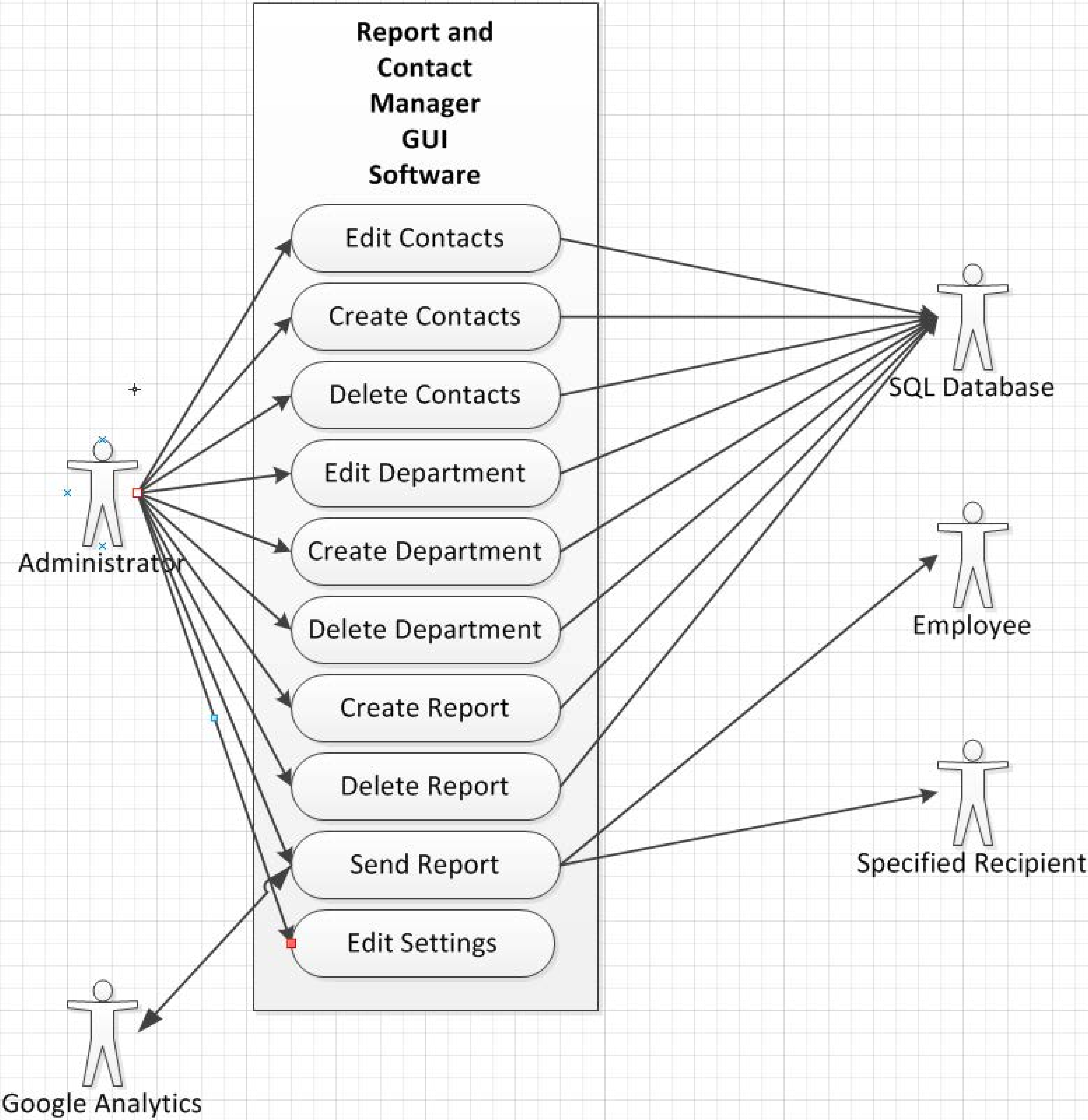
1. Respond to inbound changes from **MailSenderClass** to email related settings.
2. Return email related settings to **MailSenderClass**.

Collaboration:

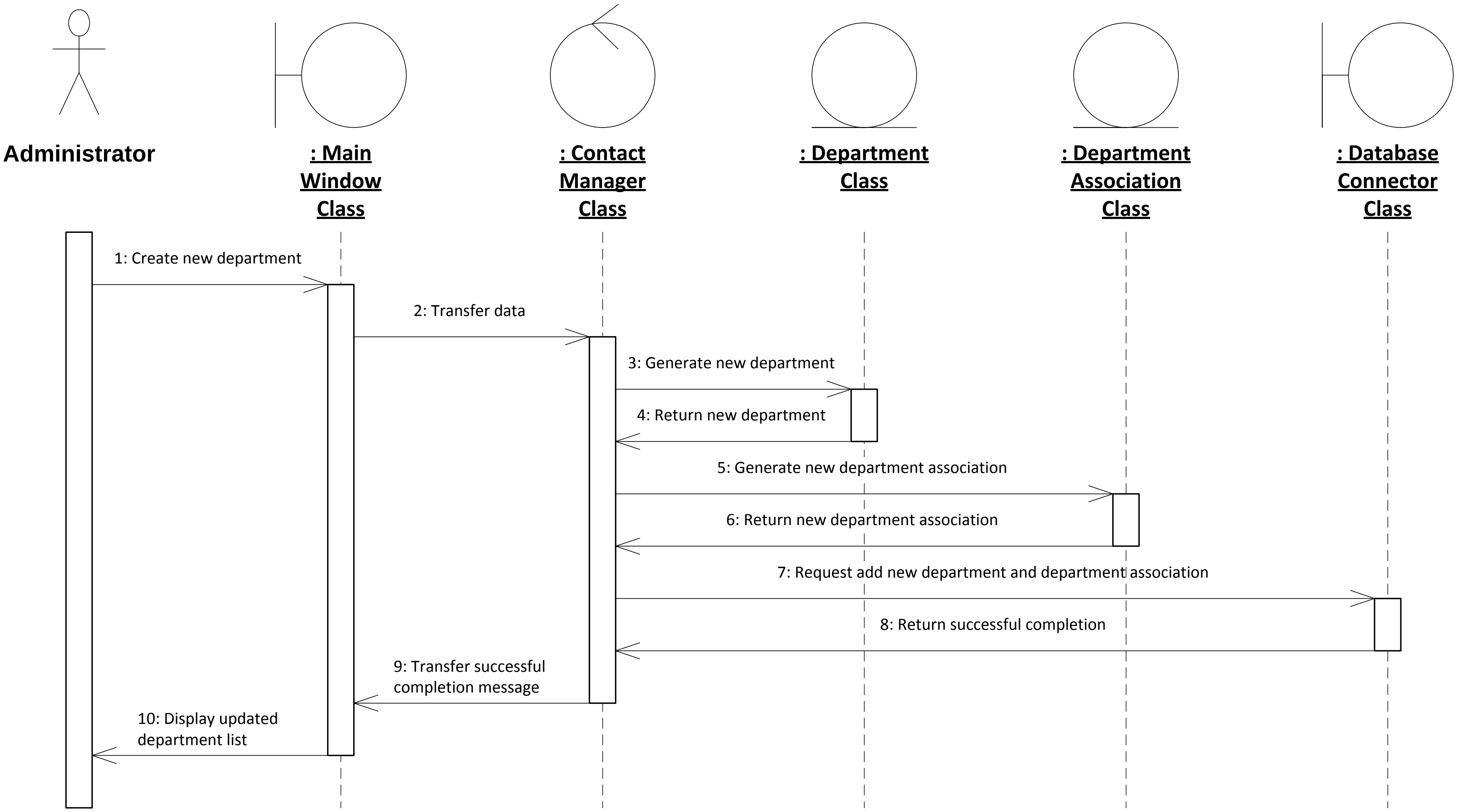
1. **MailSenderClass**


MainWindowClass
-instanceUUID : string(idl)
ContactManagerClass
-Contacts : ContactClass
-Departments : DepartmentClass
-ContactsToDepartmentsList : DepartmentAssociationClass
ReportGeneratorClass
-Reports : ReportClass
Communicates with
1
1
*
-End1
1
1
Communicates with
Communiates
1
1
DepartmentClass
-departmentID : long(idl)
-Department Name : string(idl)
-Department Telephone : string(idl)
-Department Email : string(idl)
MailManagerClass
-OutboundMessages : EmailMessageClass
-MailSender : MailerSenderClass
DatabaseConnectorClass
-DatabaseSettings : DatabaseSettingsClass
1
1
Communicates
Generates
Uses
Uses
1
*
Generates
*
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)
communicates
EmailMessageClass
-toEmail : string(idl)
-fromEmail : string(idl)
-subject : string(idl)
-contents : string(idl)
AnalyticsConnectorClass
-Result : string(idl)
*
1
Uses
communicates
*
-End2
ReportClass
-reportId : unsigned long(idl)
-Title : string(idl)
-StartDate : string(idl)
-EndDate : string(idl)
-Dimensions : string(idl)
-Metrics : string(idl)
-Filters : string(idl)
DepartmentAssociationClass
-departmentId : long(idl)
-employeeId : long(idl)
0..*
0..*
1
1
MailerSenderClass
-MailSenderSettings : MailSenderSettingsClass
SettingsManagerClass
-MailSenderSettings : MailSenderSettingsClass
-DatabaseSettings : DatabaseSettingsClass
1
Uses
1
MailSenderSettingsClass
-EmailHost : string(idl)
-userName : string(idl)
-password : string(idl)
-port : unsigned long(idl)
-isAuthRequired : boolean(idl)
-encryptionMethod : string(idl)
Contains
DatabaseSettingsClass
-databaseType : string(idl)
-host : string(idl)
-username : string(idl)
-password : string(idl)
-databaseName : string(idl)
-port : unsigned long(idl)
Contains
1
1

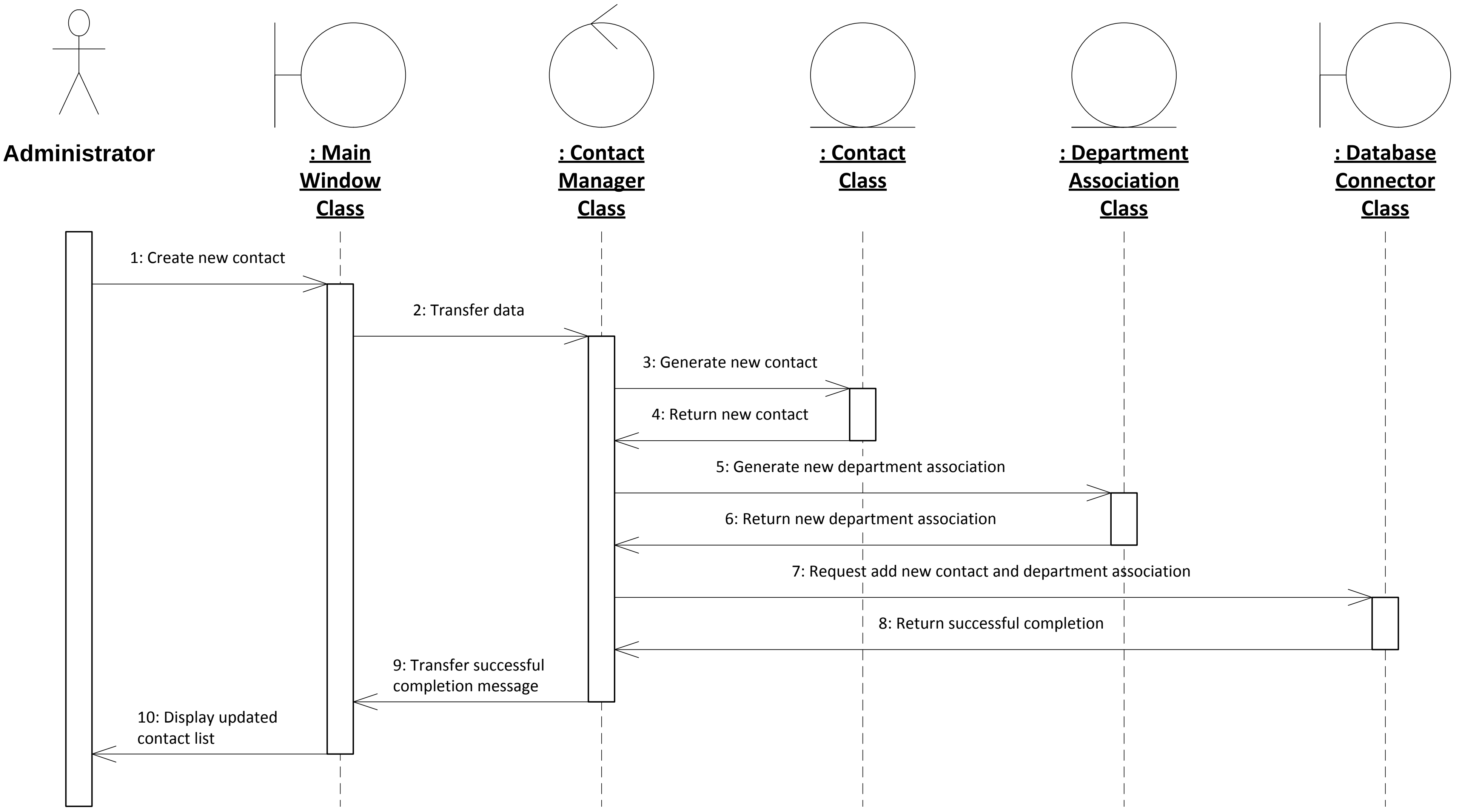
Use-Case Diagram



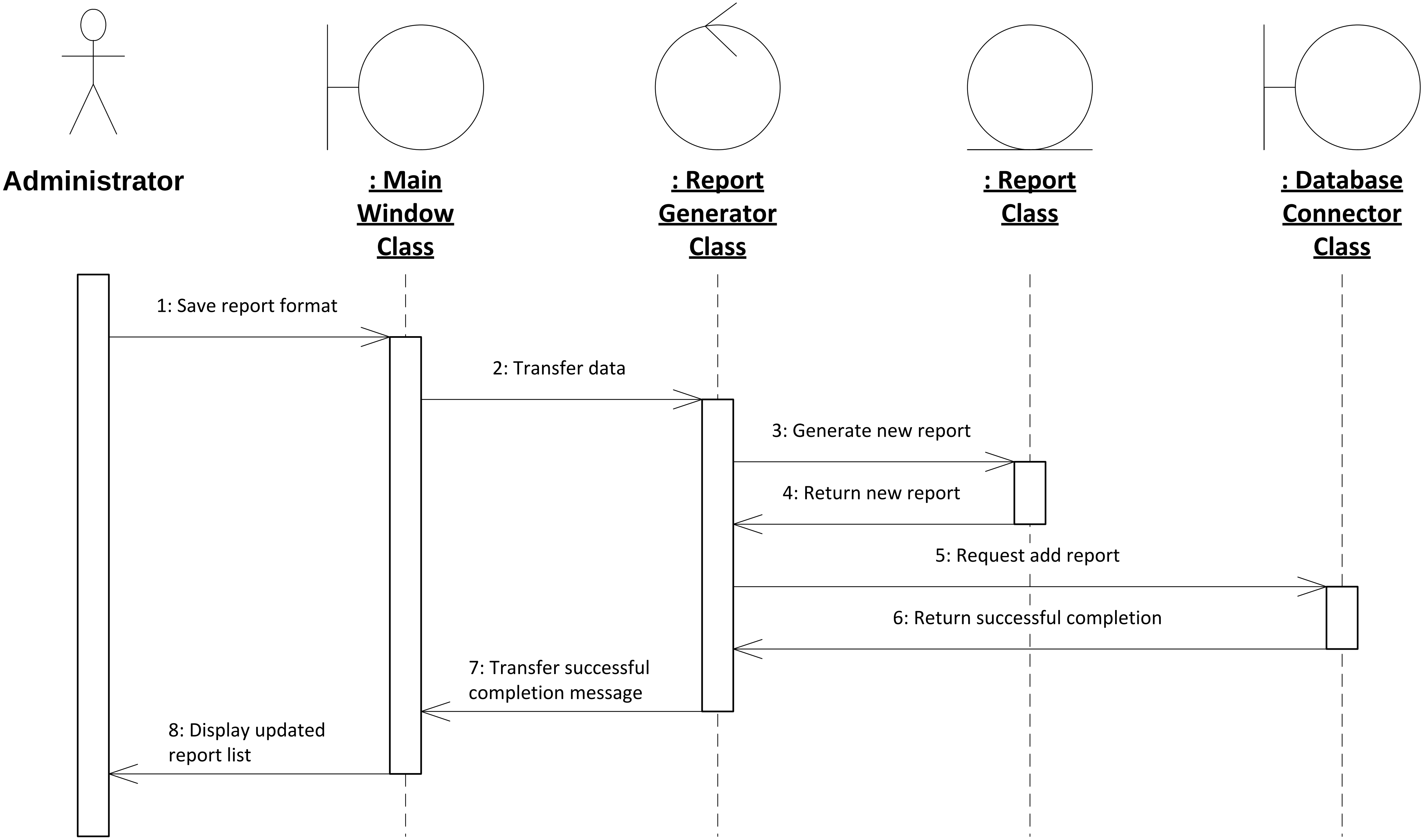
CreateDepartment


Administrator
: Main Window Class
: Contact Manager Class
: Department Class
: Department Association Class
: Database Connector Class
1: Create new department
2: Transfer data
3: Generate new department
4: Return new department
5: Generate new department association
6: Return new department association
7: Request add new department and department association
8: Return successful completion
9: Transfer successful completion message
10: Display updated department list

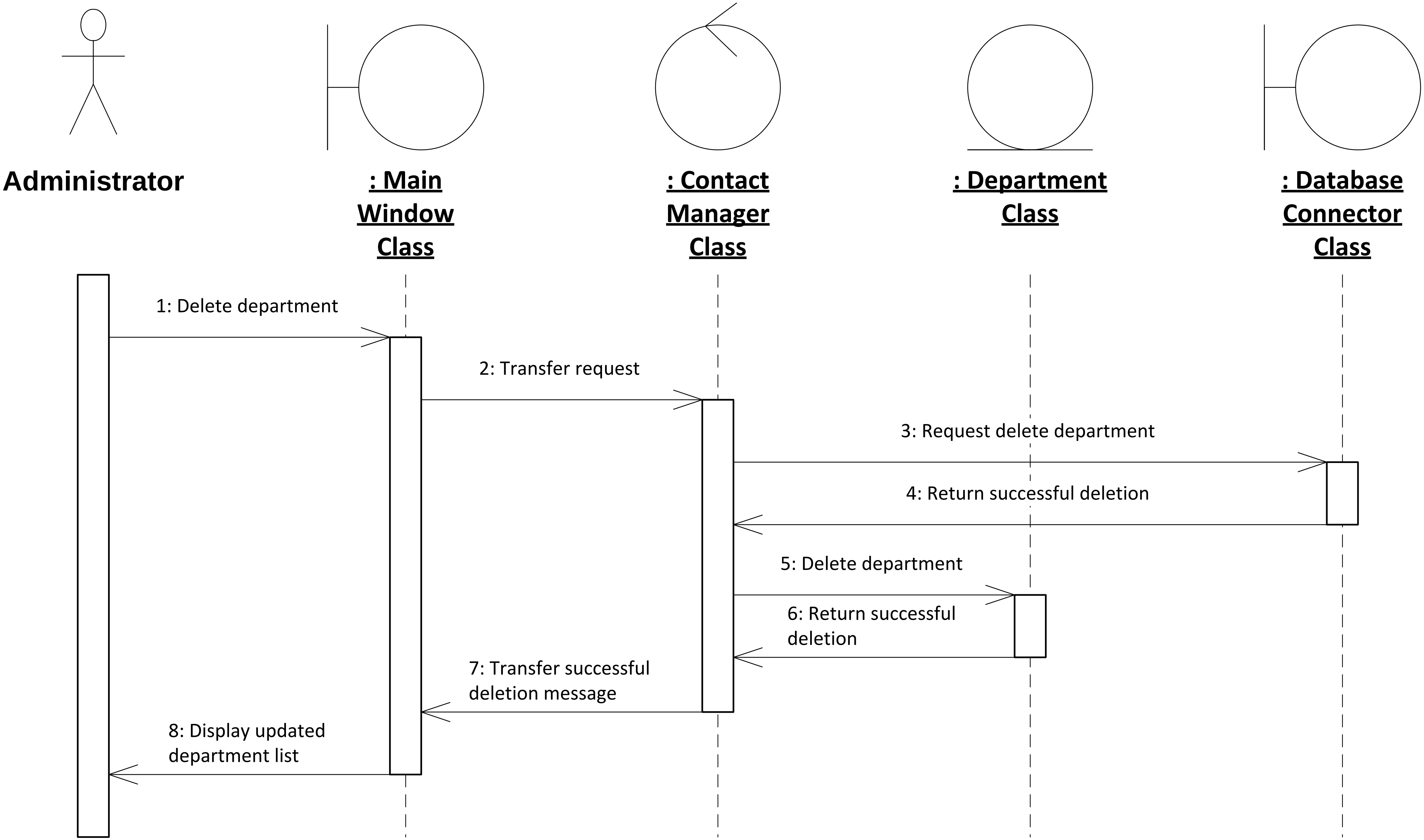
CreateEmployee


Administrator
: Main Window Class
: Contact Manager Class
: Contact Class
: Department Association Class
: Database Connector Class
1: Create new contact
2: Transfer data
3: Generate new contact
4: Return new contact
5: Generate new department association
6: Return new department association
7: Request add new contact and department association
8: Return successful completion
9: Transfer successful completion message
10: Display updated contact list

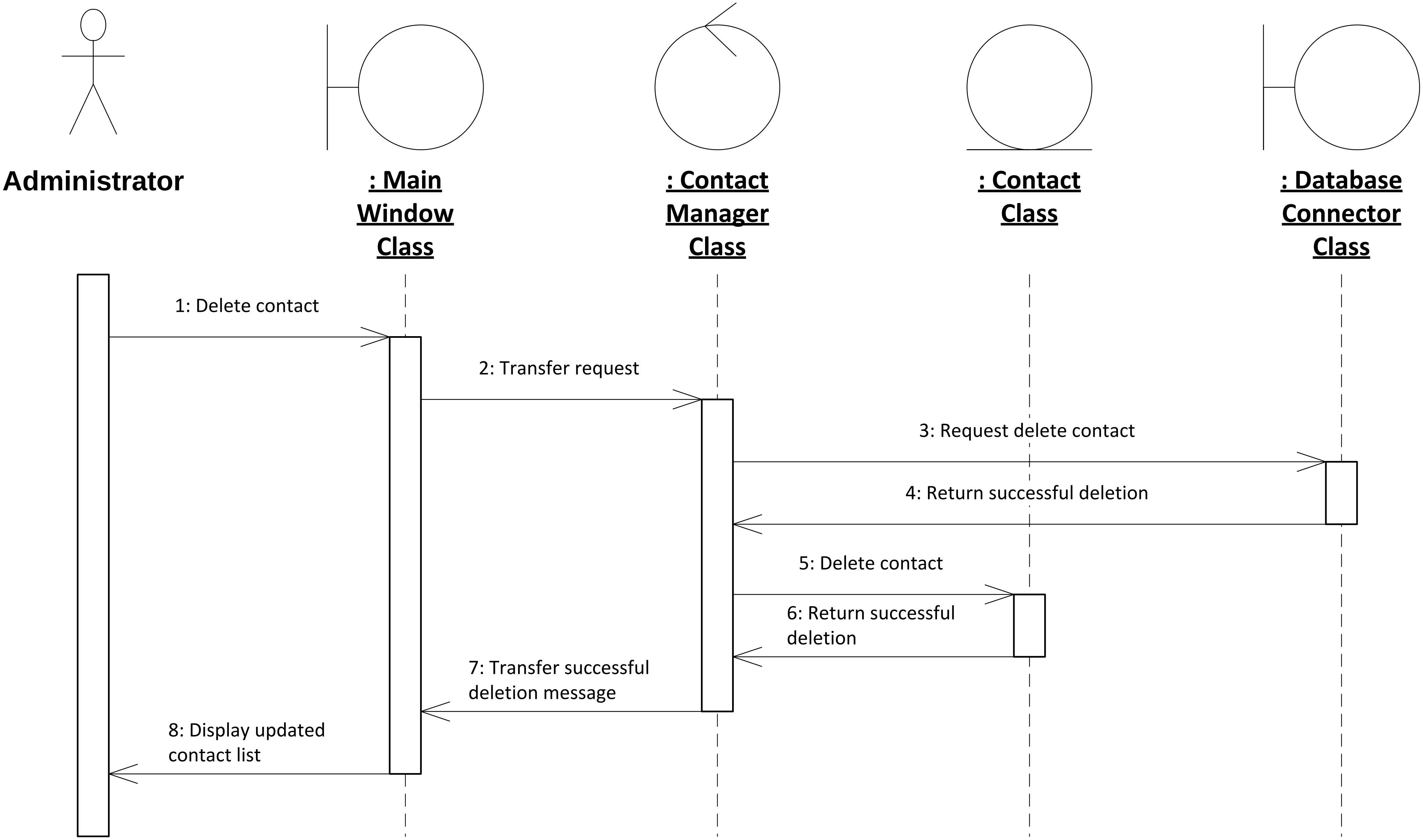
CreateReport



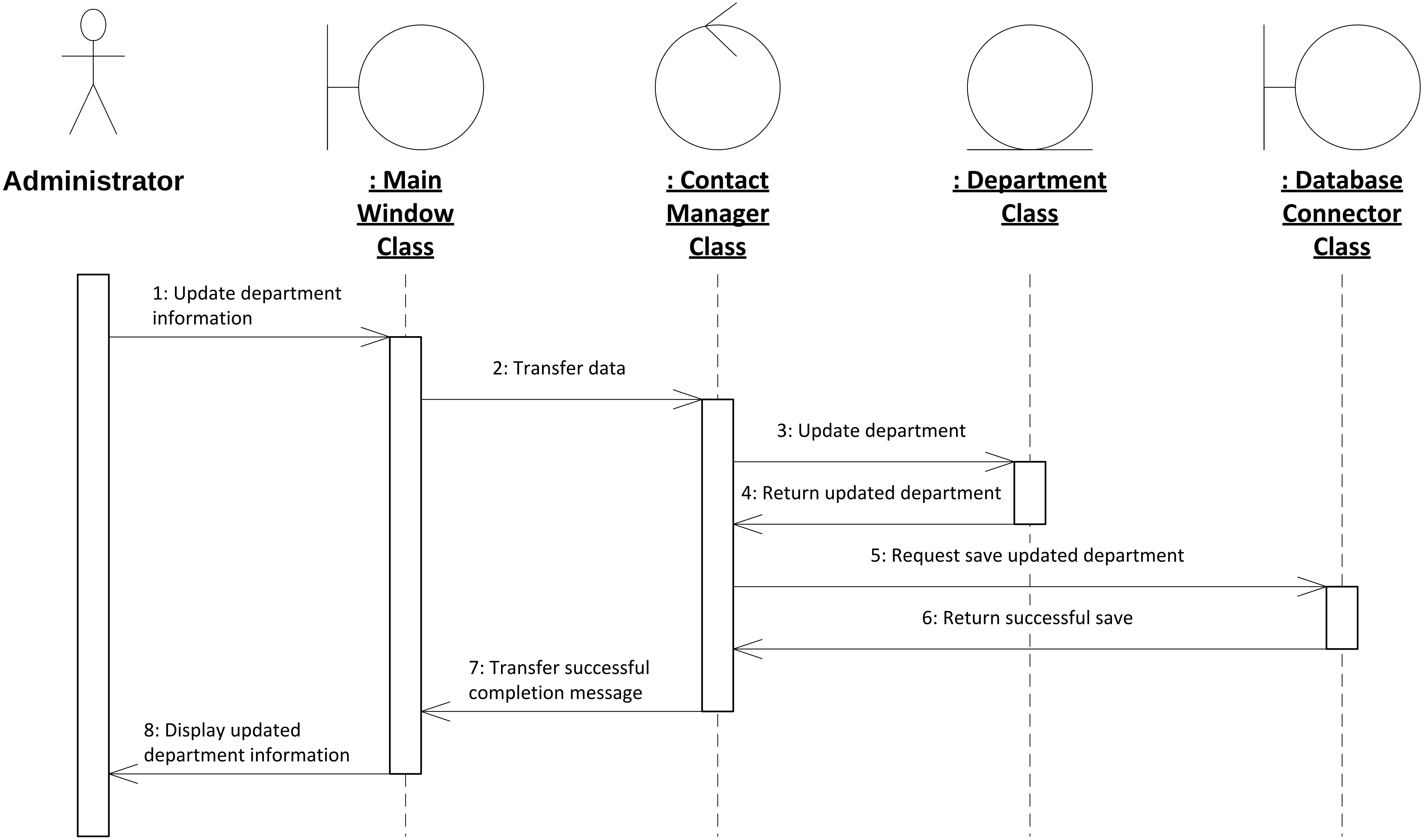
DeleteDepartment


Administrator
: Main Window Class
: Contact Manager Class
: Department Class
: Database Connector Class
1: Delete department
2: Transfer request
3: Request delete department
4: Return successful deletion
5: Delete department
6: Return successful deletion
7: Transfer successful deletion message
8: Display updated department list

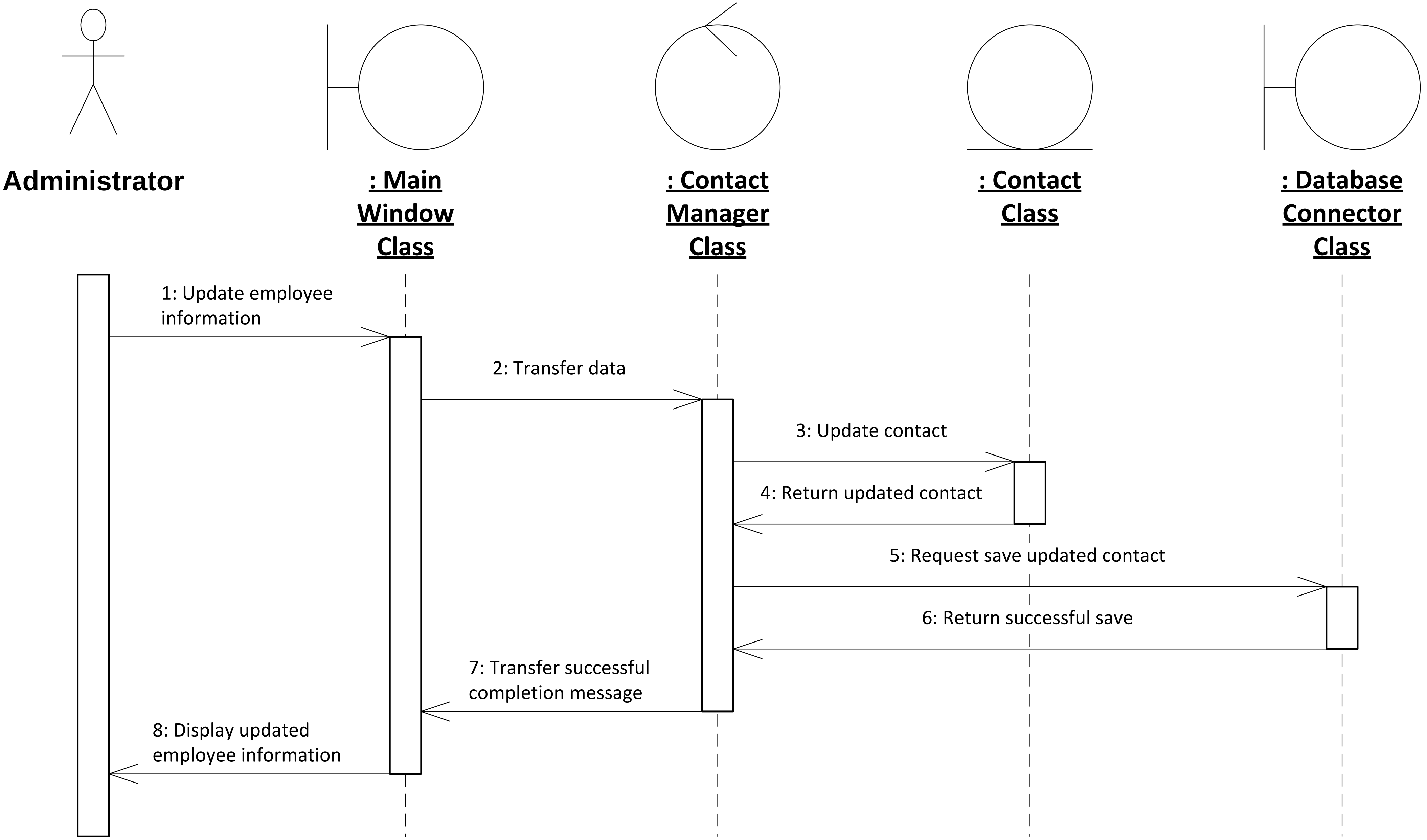
DeleteEmployee


Administrator
: Main Window Class
: Contact Manager Class
: Contact Class
: Database Connector Class
1: Delete contact
2: Transfer request
3: Request delete contact
4: Return successful deletion
5: Delete contact
6: Return successful deletion
7: Transfer successful deletion message
8: Display updated contact list

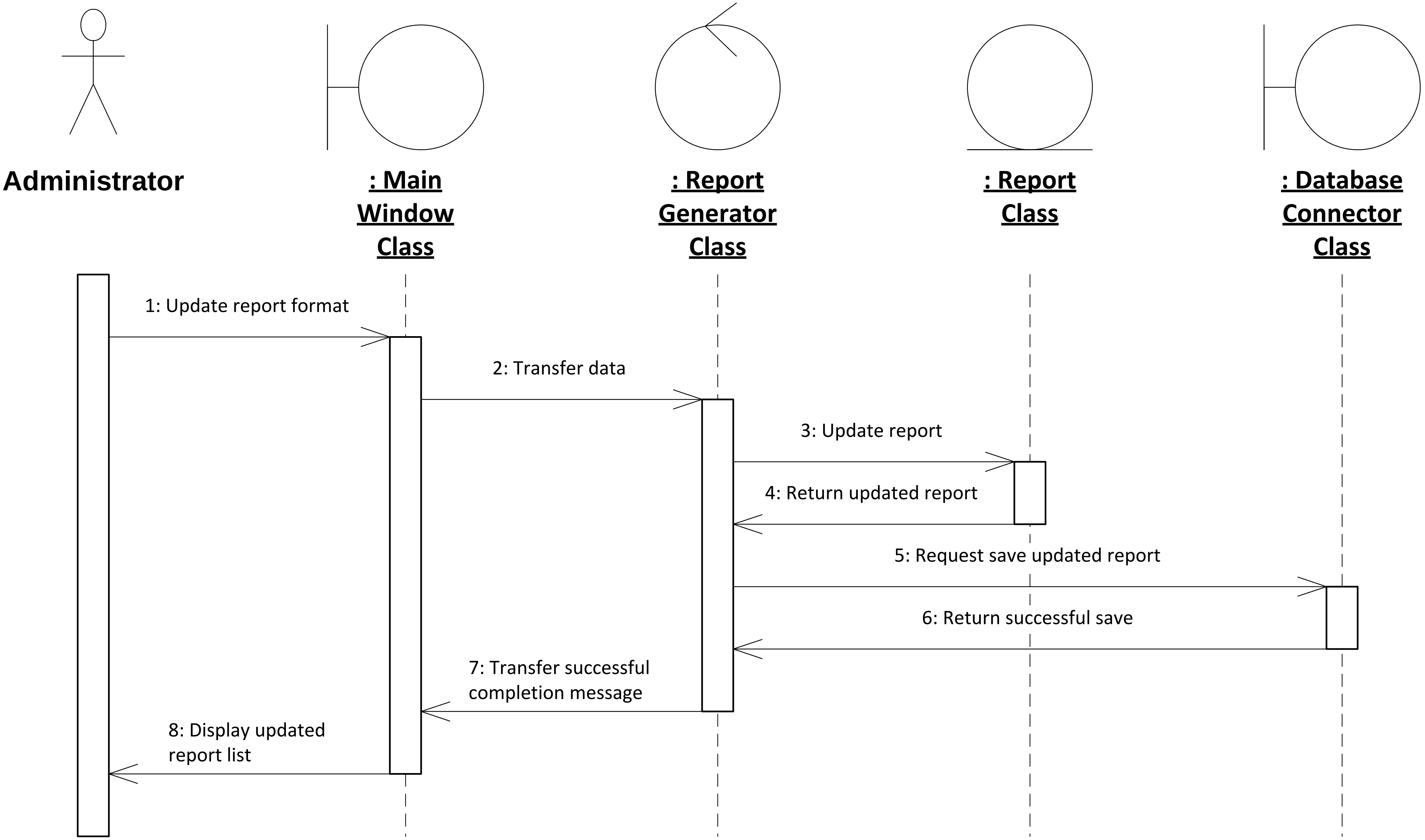
EditDepartment


Administrator
: Main Window Class
: Contact Manager Class
: Department Class
: Database Connector Class
1: Update department information
2: Transfer data
3: Update department
4: Return updated department
5: Request save updated department
6: Return successful save
7: Transfer successful completion message
8: Display updated department information

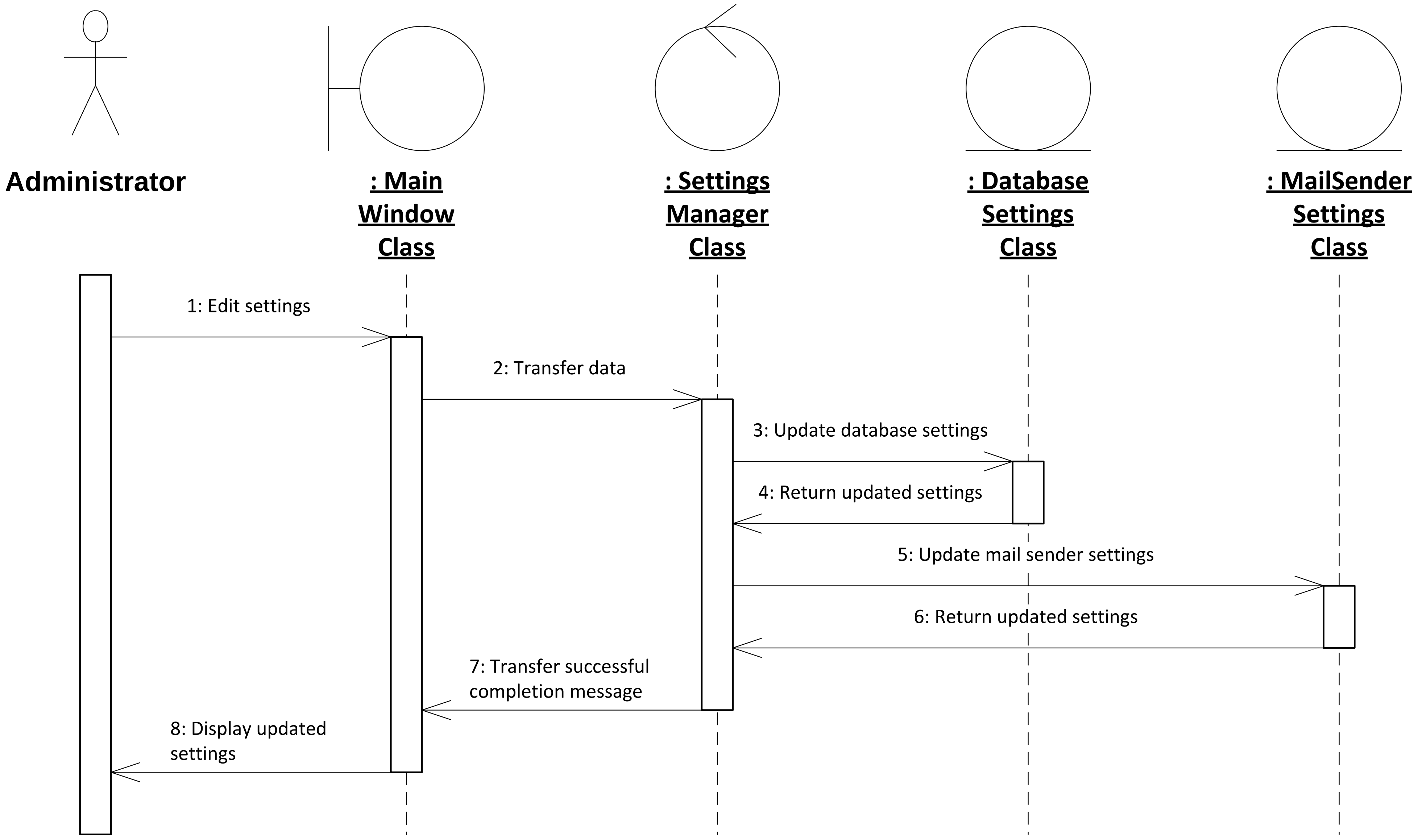
EditEmployee


Administrator
: Main Window Class
: Contact Manager Class
: Contact Class
: Database Connector Class
1: Update employee information
2: Transfer data
3: Update contact
4: Return updated contact
5: Request save updated contact
6: Return successful save
7: Transfer successful completion message
8: Display updated employee information

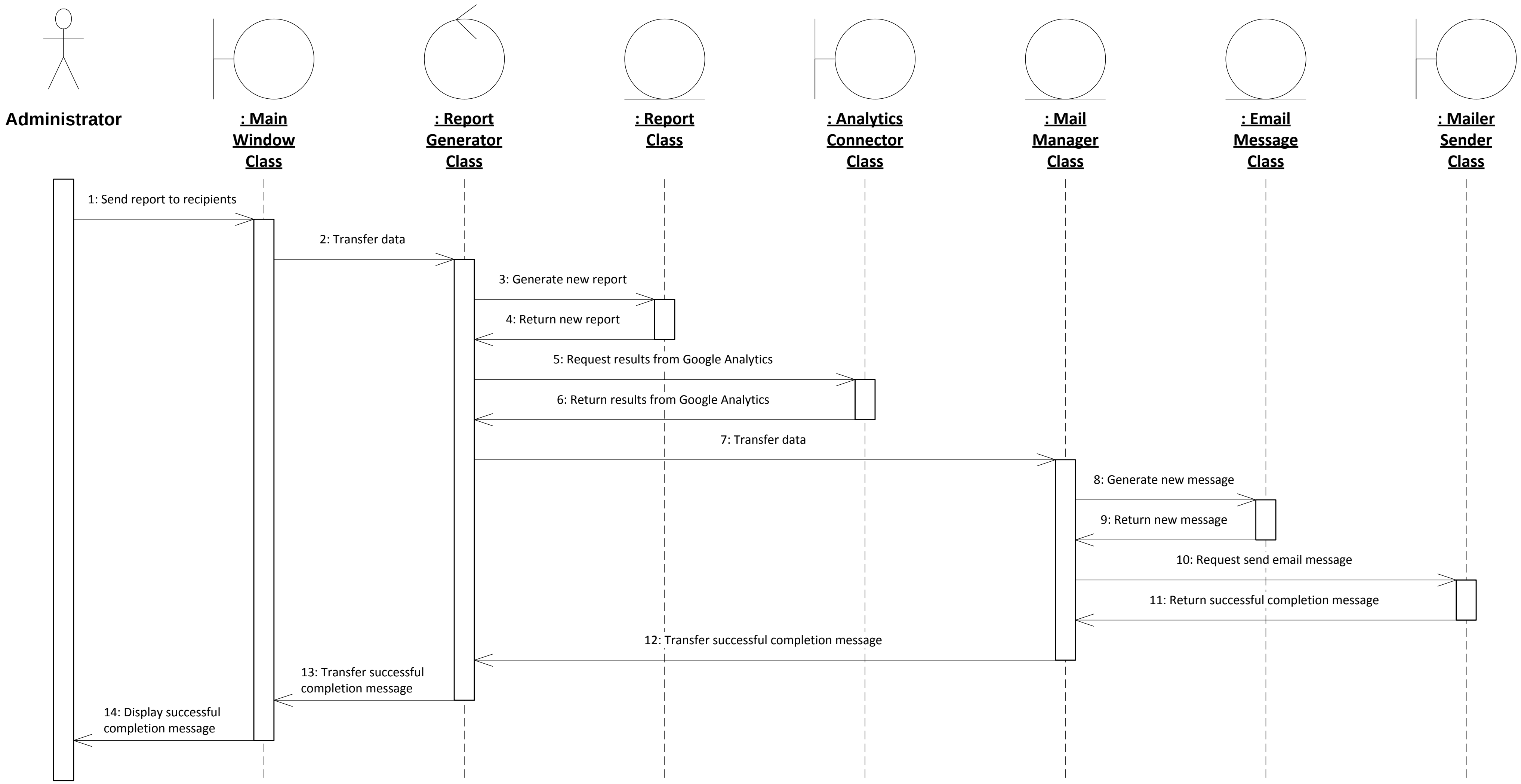
EditReport


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

EditSettings


Administrator
: Main Window Class
: Settings Manager Class
: Database Settings Class
: MailSender Settings Class
1: Edit settings
2: Transfer data
3: Update database settings
4: Return updated settings
5: Update mail sender settings
6: Return updated settings
7: Transfer successful completion message
8: Display updated settings

SendReport

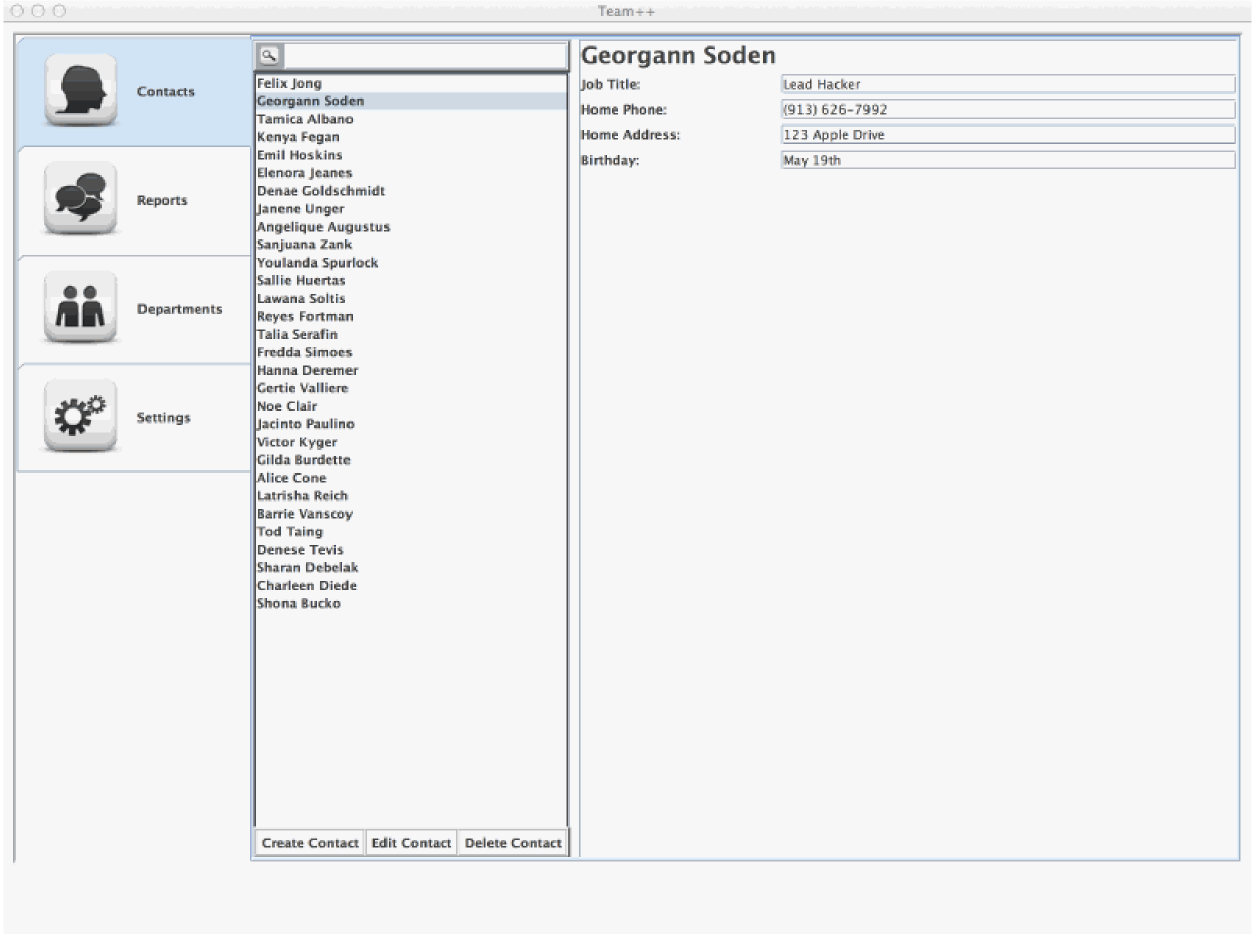

Administrator
: Main Window Class
: Report Generator Class
: Report Class
: Analytics Connector Class
: Mail Manager Class
: Email Message Class
: Mailer Sender Class
1: Send report to recipients
2: Transfer data
3: Generate new report
4: Return new report
5: Request results from Google Analytics
6: 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

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.

GUI Descriptions:

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

Contacts Tab:



Felix Jong
Georgann Soden
Tamica Albano
Kenya Fegan
Emil Hoskins
Elenora Jeanes
Denae Goldschmidt
Janene Unger
Angelique Augustus
Sanjuana Zank
Youlanda Spurlock
Sallie Huertas
Lawana Soltis
Reyes Fortman
Talia Serafin
Fredda Simoes
Hanna Deremer
Gertie Valliere
Noe Clair
Jacinto Paulino
Victor Kyger
Gilda Burdette
Alice Cone
Latrisha Reich
Barrie Vanscoy
Tod Taing
Denese Tevis
Sharan Debelak
Charleen Diede
Shona Bucko

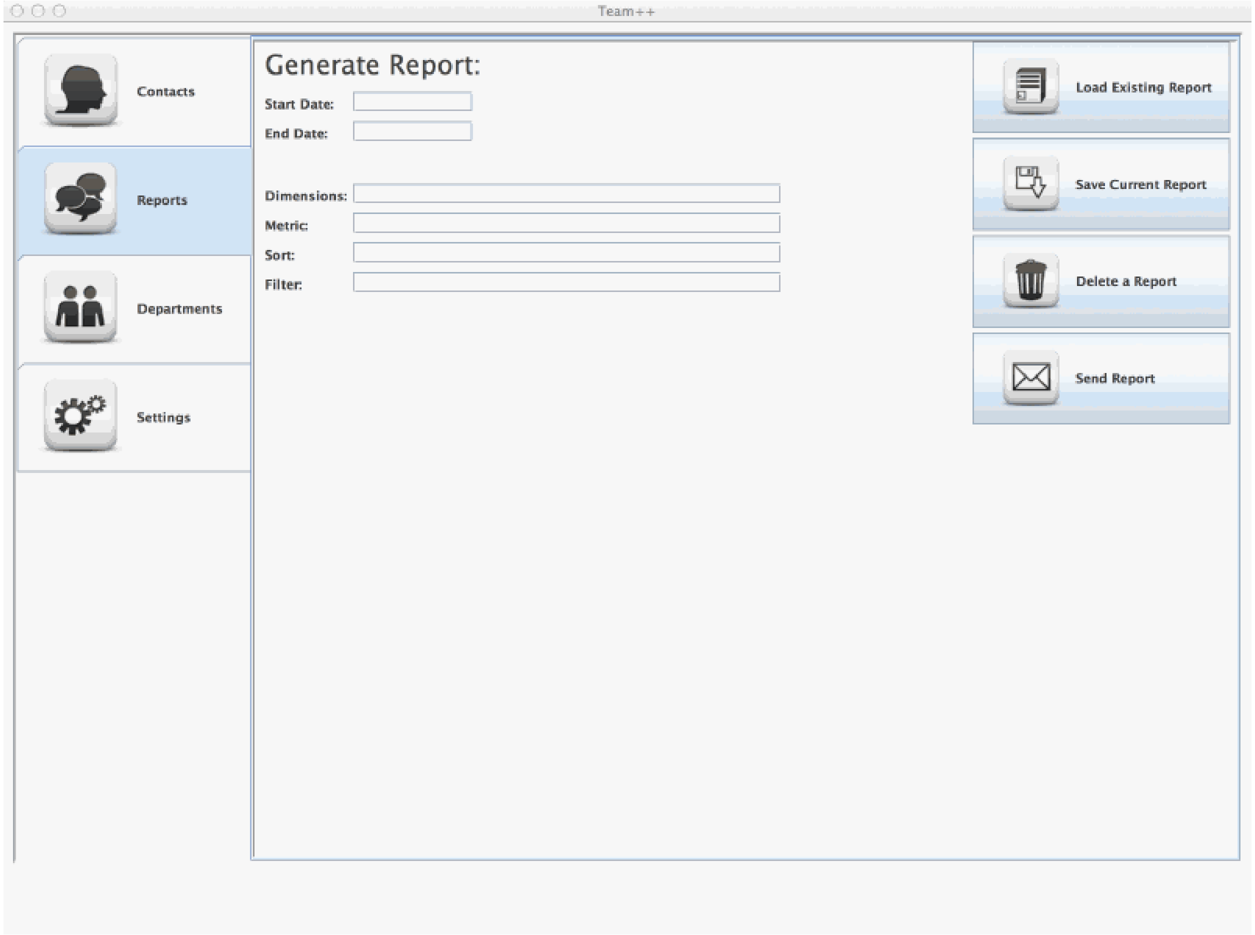
Georgann Soden

Job Title:	Lead Hacker
Home Phone:	(913) 626-7992
Home Address:	123 Apple Drive
Birthday:	May 19th

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:

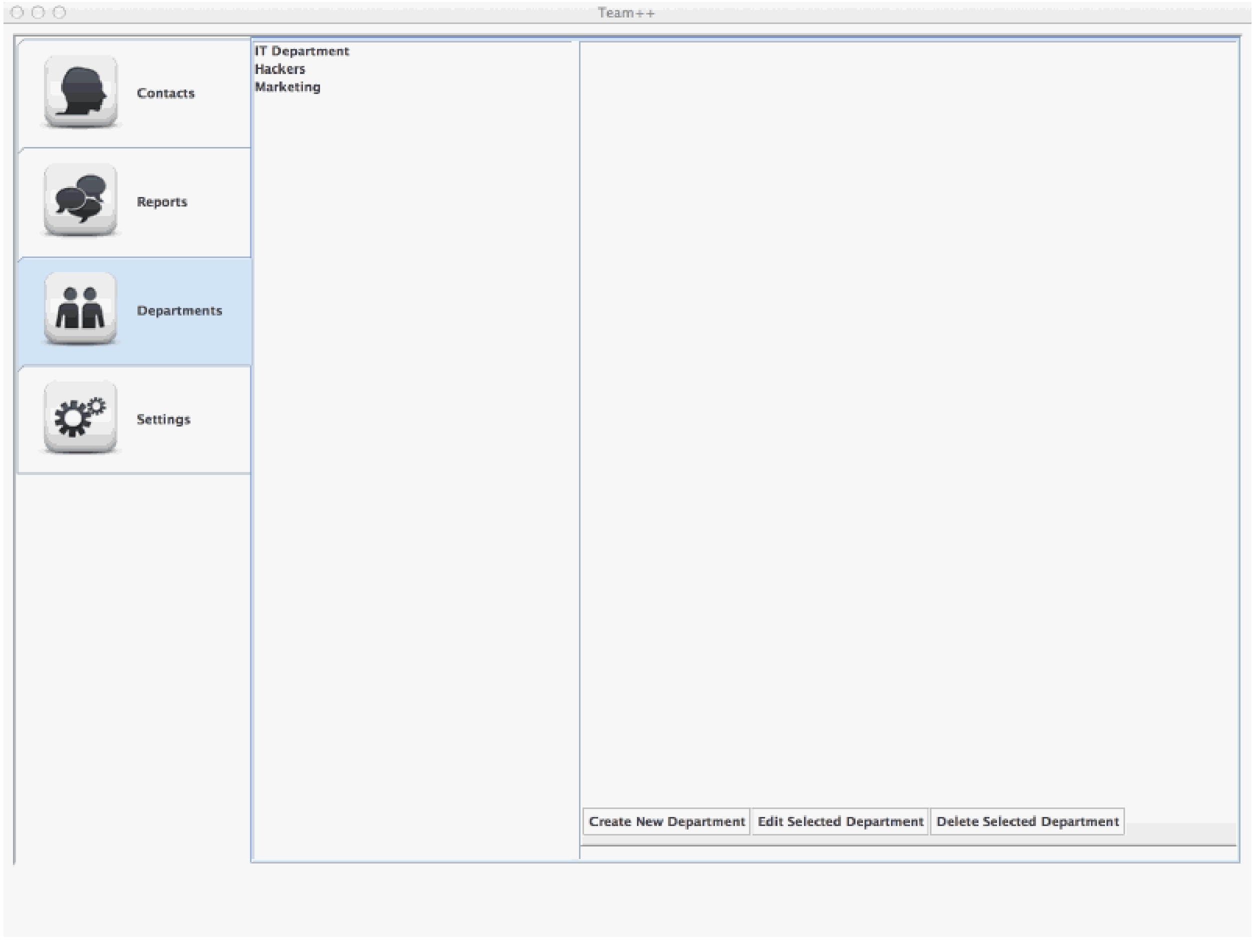
- Start Date
- End Date
- Dimensions
- Metric
- Sort
- Filter

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:

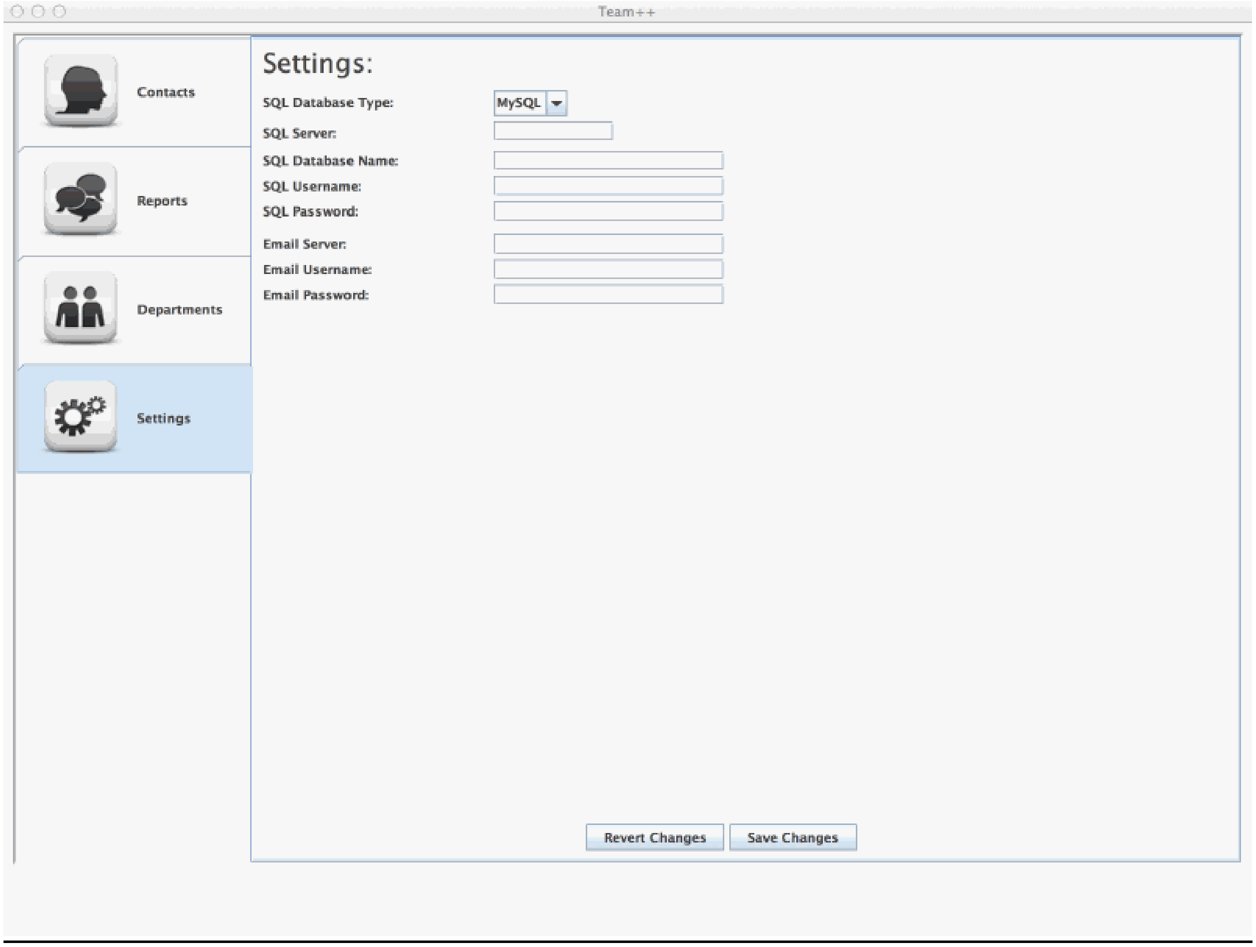


IT Department
Hackers
Marketing

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:



Settings:

SQL Database Type: MySQL
SQL Server:
SQL Database Name:
SQL Username:
SQL Password:
Email Server:
Email Username:
Email Password:

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.