Software Design and Architecture

Phase 3

Group 13: Devante Wilson - 100554361

Shahrukh Zarir - 100489271

Pranav Yadav - 100557540

Tsering Paljor - 100521258

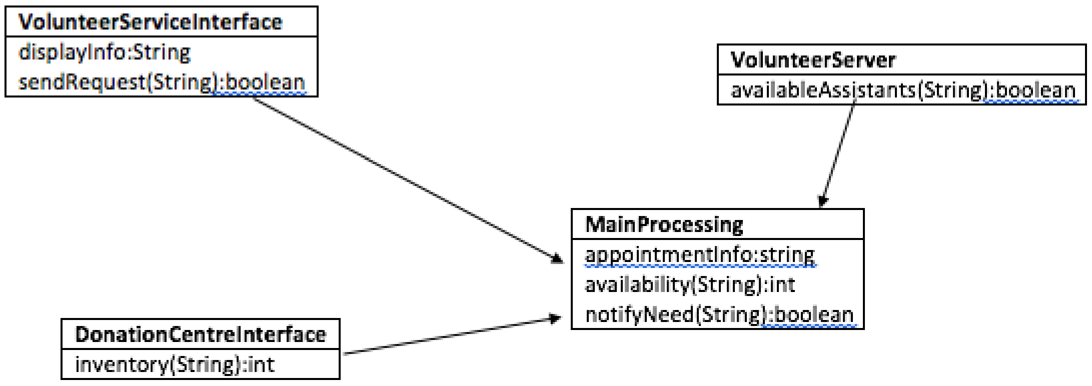
Date: 12/5/2016

Overview & Purpose

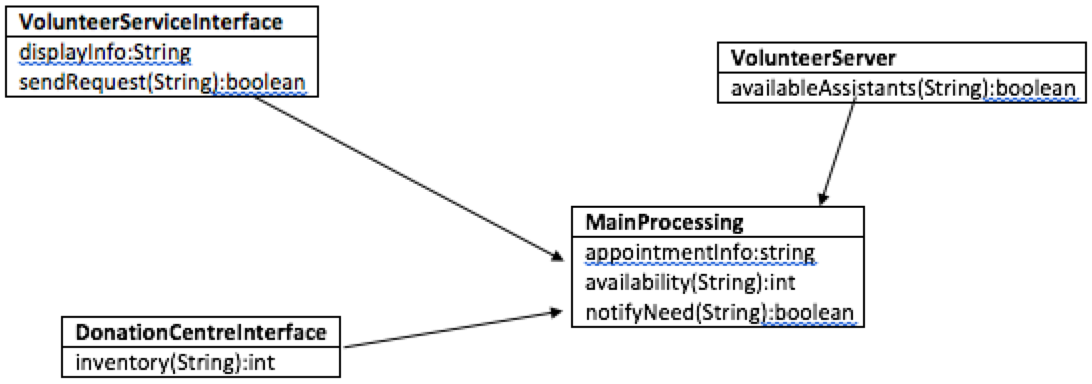
This part of the project finalizes the iVolunteer system. By developing more detailed descriptions and diagrams, our team is able to improve the overall quality of the design.

Class Diagrams

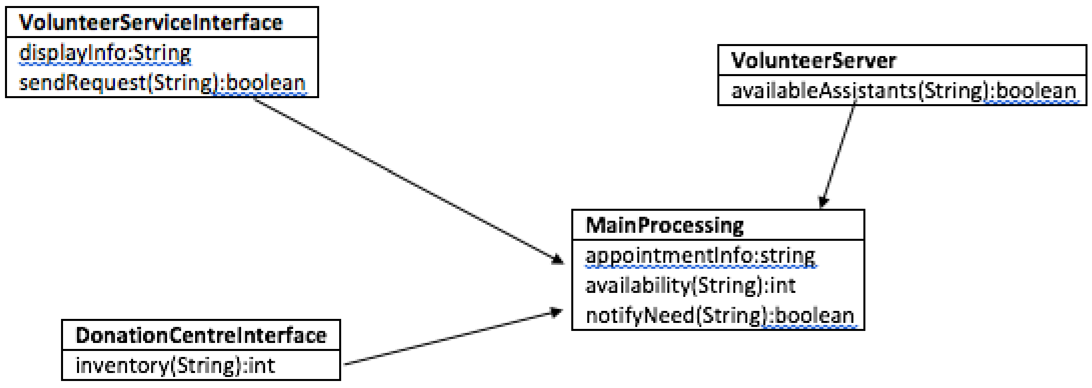
Delivery Service Class Diagram



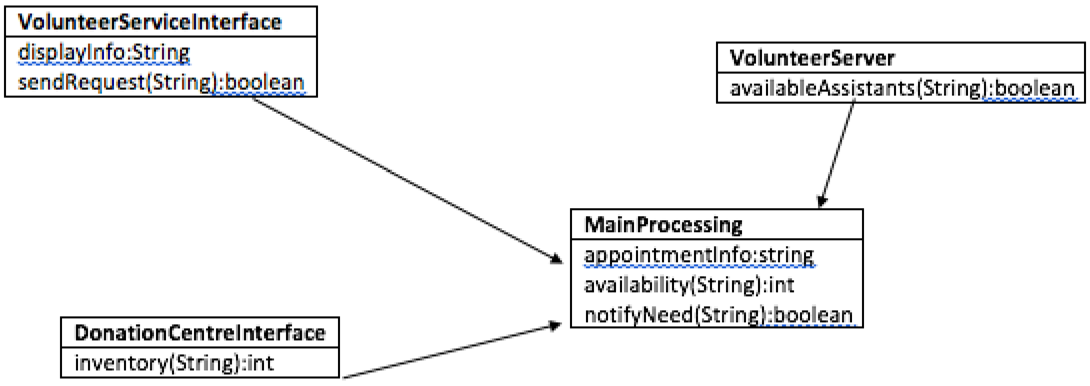
Financial Planning Class Diagram



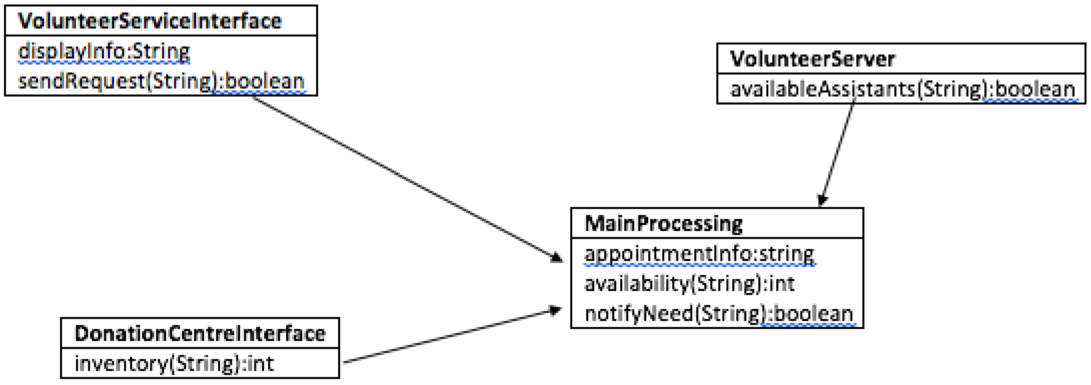
Food Delivery Class Diagram



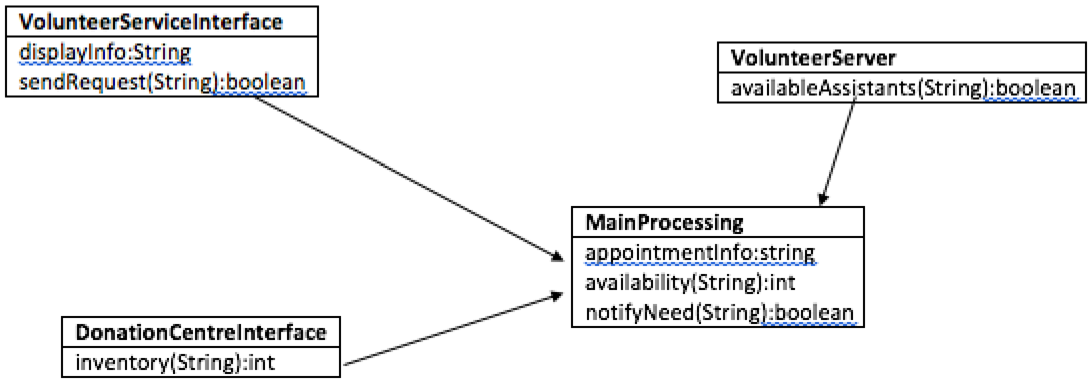
Child Care Class Diagram



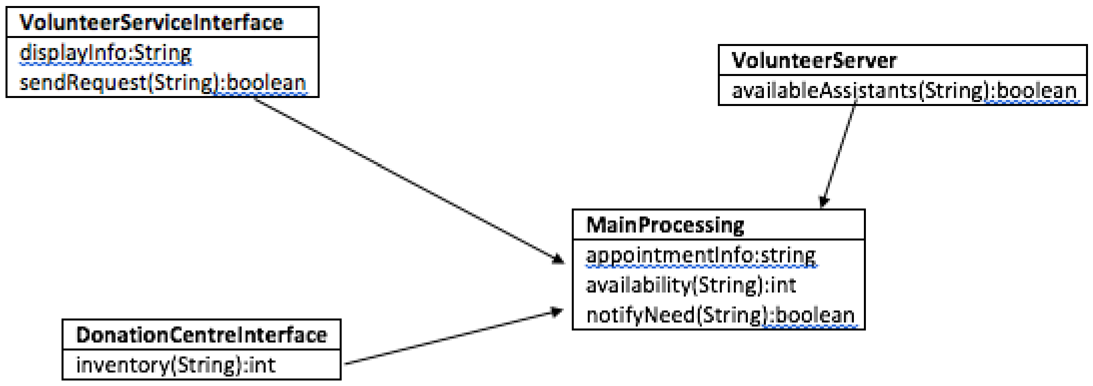
Hygiene Care Class Diagram



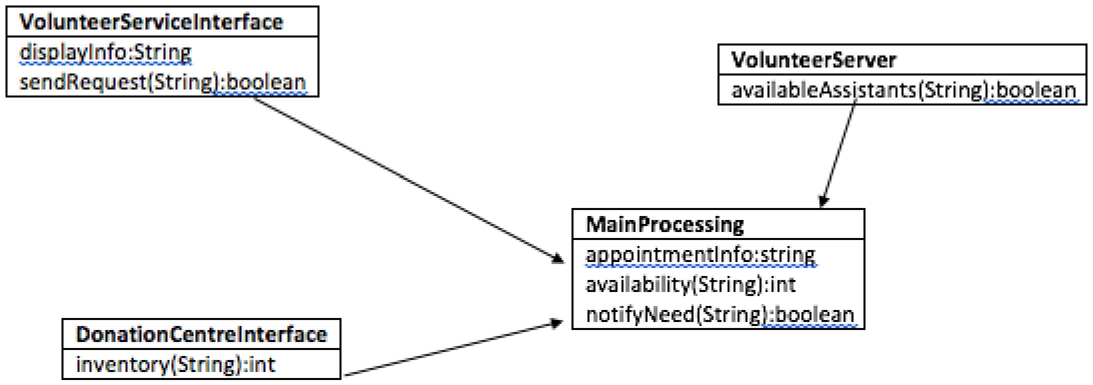
Health Care Class Diagram



Blood Donation Class Diagram



Immunization Class Diagram



Class Interface Specification

VolunteerServiceInterface

sendRequest()

Function performed: handle sending and receiving requests from/to the beneficiary.

Precondition: beneficiary must want a type of service offered.

Postcondition: verdict must be boolean (true).

Invariant:

Input parameters: String serviceType (type of service needed)

Output parameters: boolean (verdict)

Operations used from others:

displayInfo()

Function performed: display final information to the beneficiary.

Precondition: there must be service information to display (appointment information or service not available)

Postcondition: Some sort of service information is returned

Invariant: service was requested

Input parameters: String serviceInfo (appointment information or service not available)

Output parameters: String (service details)

Operations used from others: MainProcessing.giveServiceDetails()

MainProcessing

giveServiceDetails()

Function performed: specify appointment information or if the service is not available

Precondition: there is service information to return

Postcondition: some sort of service information is returned

Invariant: service was requested

Input parameters: boolean volunteerVerdict (can volunteer provide service?)

Output parameters: String (appointment information or service not available)

Operations used from others: VolunteerServer.volunteerAvailable()

getAvailability()

Function performed: send request to VolunteerServer and DonationCentreInterface to check for assistant and supply availability

Precondition: service request valid

Postcondition:

Invariant: service was requested

Input parameters: String serviceType (type of service requested)

Output parameters: int (how many volunteers)

Operations used from others:

notifyOfNeed()

Function performed: notify volunteer of need

Precondition: Valid volunteer found from VolunteerServer. Supplies also need to be available for the them to use (query DonationCentreInterface)

Postcondition: volunteer has to be available to deliver service.

Invariant: service was requested

Input parameters: String serviceType (type of service requested)

Output parameters: boolean (can volunteer provide service?)

Operations used from others: VolunteerServer.volunteerAvailable()

DonationCentreInterface

inventoryAvailable()

Function performed: process inventory information for items sent to the donation centre from the wealth giver.

Precondition:

Postcondition: if inventory available (or none) that verdict will be returned

Invariant: service was requested

Input parameters: String serviceType (type of service to be provided)

Output parameters: boolean (is there inventory available?)

Operations used from others:

VolunteerServer

volunteerAvailable()

Function performed: process available assistants who fit the beneficiary request

Precondition:

Postcondition: all valid volunteers (or none) will be returned

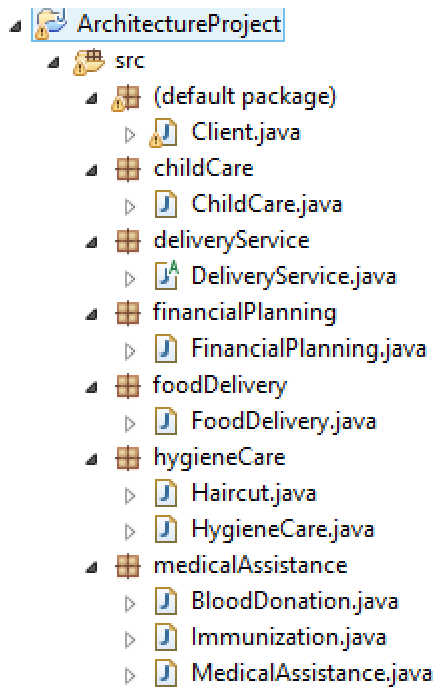
Invariant: service was requested

Input parameters: String serviceType (type of service to be provided)

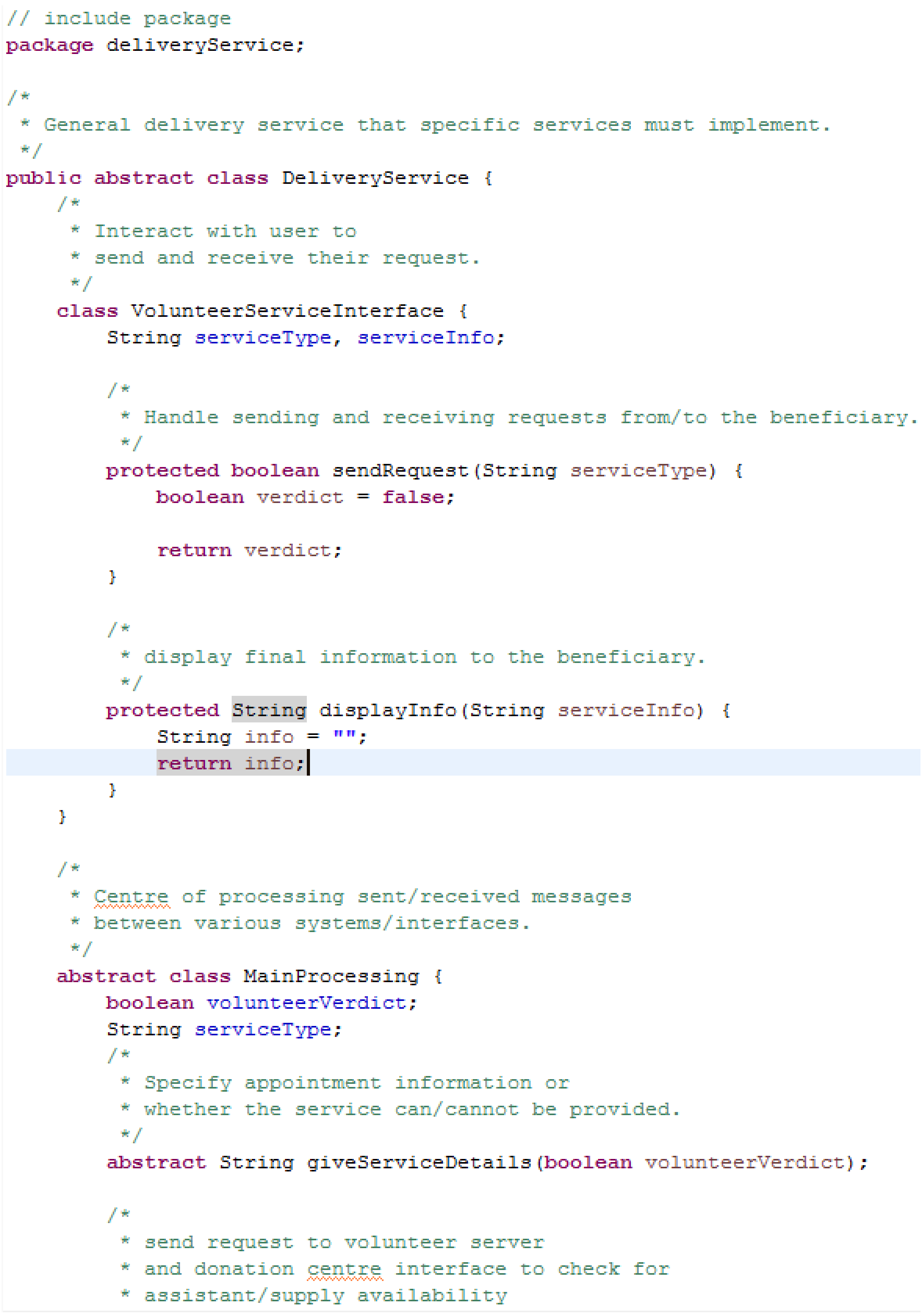
Output parameters: boolean (verdict - any assistants available?)

Operations used from others:

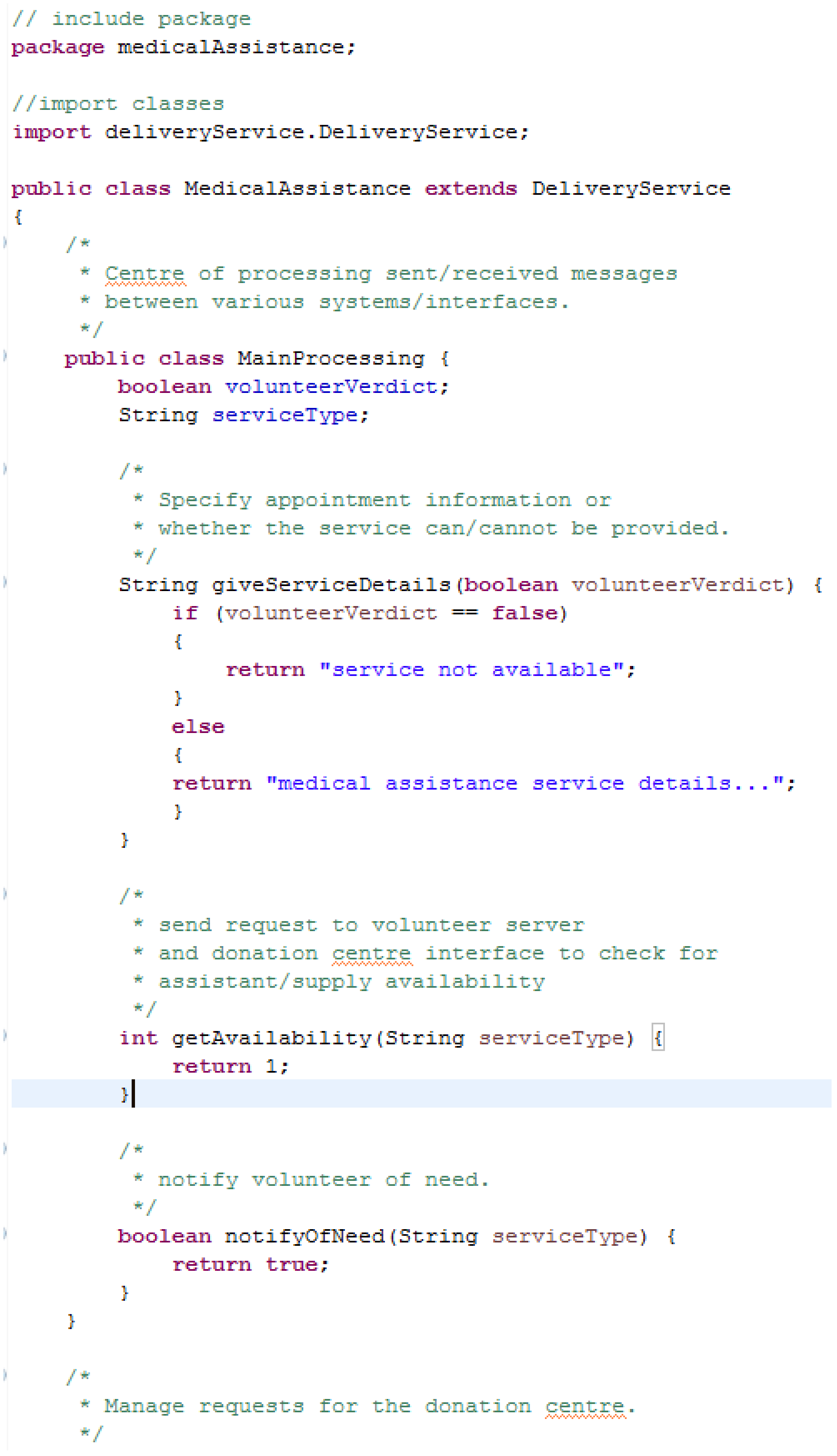
Implementation



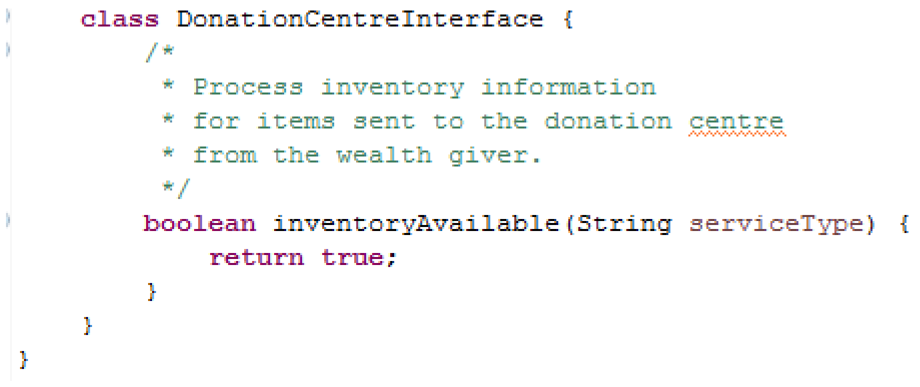
```
ArchitectureProject
  src
    (default package)
      Client.java
    childCare
      ChildCare.java
    deliveryService
      DeliveryService.java
    financialPlanning
      FinancialPlanning.java
    foodDelivery
      FoodDelivery.java
    hygieneCare
      Haircut.java
      HygieneCare.java
    medicalAssistance
      BloodDonation.java
      Immunization.java
      MedicalAssistance.java
```



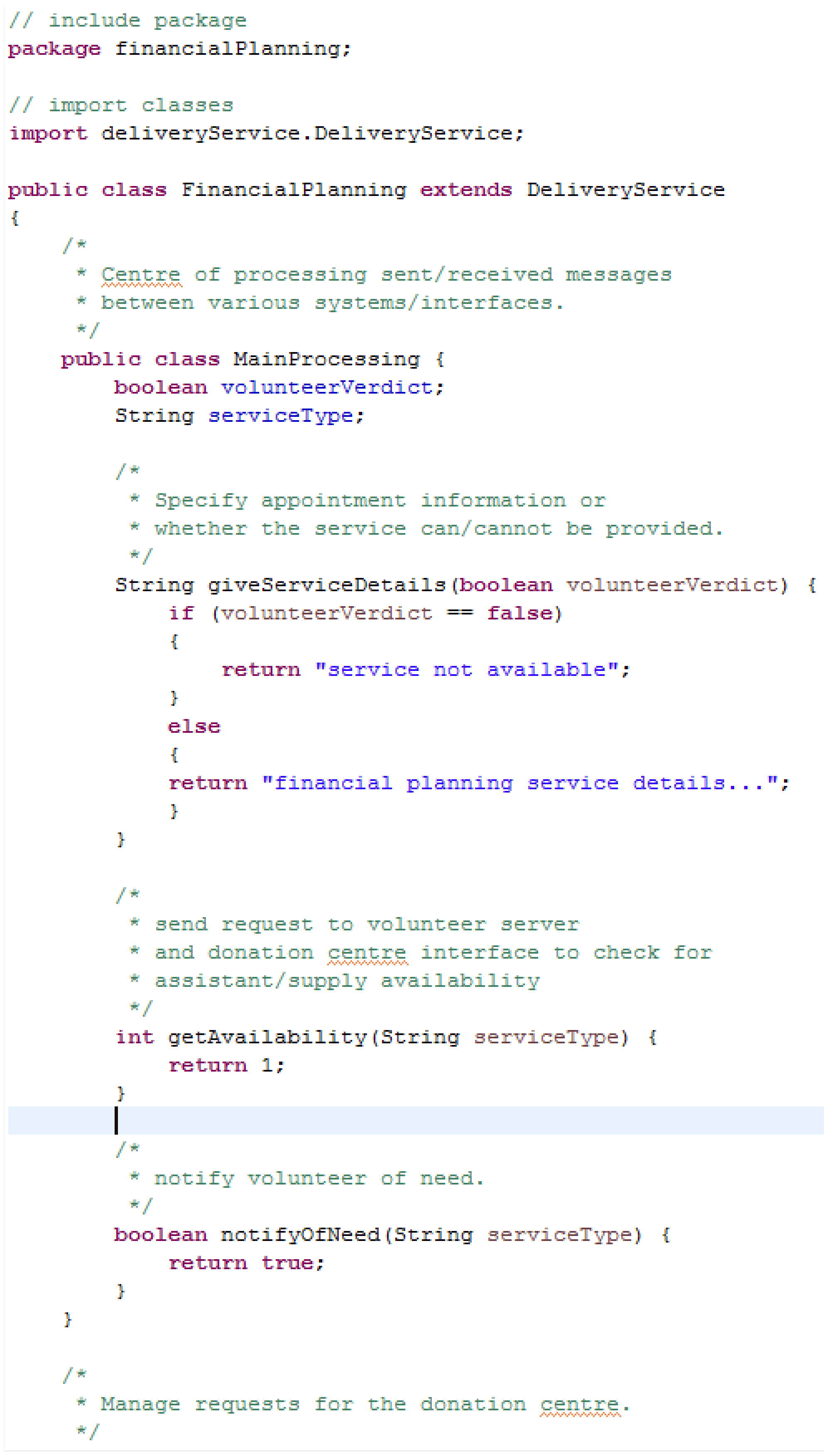
```java
// include package
package deliveryService;

/*
 * General delivery service that specific services must implement.
 */
public abstract class DeliveryService {
    /*
     * Interact with user to
     * send and receive their request.
     */
    class VolunteerServiceInterface {
        String serviceType, serviceInfo;

        /*
         * Handle sending and receiving requests from/to the beneficiary.
         */
        protected boolean sendRequest(String serviceType) {
            boolean verdict = false;

            return verdict;
        }

        /*
         * display final information to the beneficiary.
         */
        protected String displayInfo(String serviceInfo) {
            String info = "";
            return info;
        }
    }

    /*
     * Centre of processing sent/received messages
     * between various systems/interfaces.
     */
    abstract class MainProcessing {
        boolean volunteerVerdict;
        String serviceType;
        /*
         * Specify appointment information or
         * whether the service can/cannot be provided.
         */
        abstract String giveServiceDetails(boolean volunteerVerdict);

        /*
         * send request to volunteer server
         * and donation centre interface to check for
         * assistant/supply availability
        abstract int getAvailability(String serviceType);

        /*
         * notify volunteer of need.
         */
        abstract boolean notifyOfNeed(String serviceType);
    }

    /*
     * Manage requests for the donation centre.
     */
    abstract class DonationCentreInterface {
        /*
         * Process inventory information
         * for items sent to the donation centre
         * from the wealth giver.
         */
        abstract boolean inventoryAvailable(String serviceType);
    }

    /*
     * Manage all the volunteers apart of the iVolunteer system.
     */
    class VolunteerServer {
        /*
         * Process available assistants who fit the beneficiary request.
         */
        protected boolean volunteerAvailable(String serviceType) {
            boolean verdict = false;

            return verdict;
        }
    }
}
```



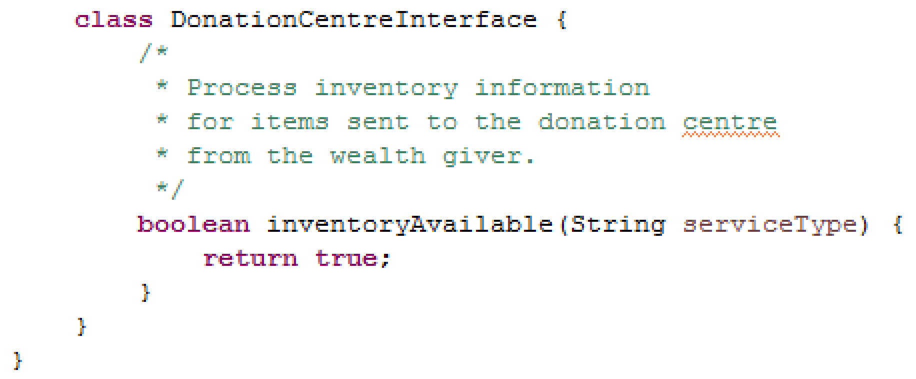
```java
// include package
package medicalAssistance;

//import classes
import deliveryService.DeliveryService;

public class MedicalAssistance extends DeliveryService
{
    /*
     * Centre of processing sent/received messages
     * between various systems/interfaces.
     */
    public class MainProcessing {
        boolean volunteerVerdict;
        String serviceType;

        /*
         * Specify appointment information or
         * whether the service can/cannot be provided.
         */
        String giveServiceDetails(boolean volunteerVerdict) {
            if (volunteerVerdict == false)
            {
                return "service not available";
            }
            else
            {
            return "medical assistance service details...";
            }
        }

        /*
         * send request to volunteer server
         * and donation centre interface to check for
         * assistant/supply availability
         */
        int getAvailability(String serviceType) {
            return 1;
        }

        /*
         * notify volunteer of need.
         */
        boolean notifyOfNeed(String serviceType) {
            return true;
        }
    }

    /*
     * Manage requests for the donation centre.
     */
```



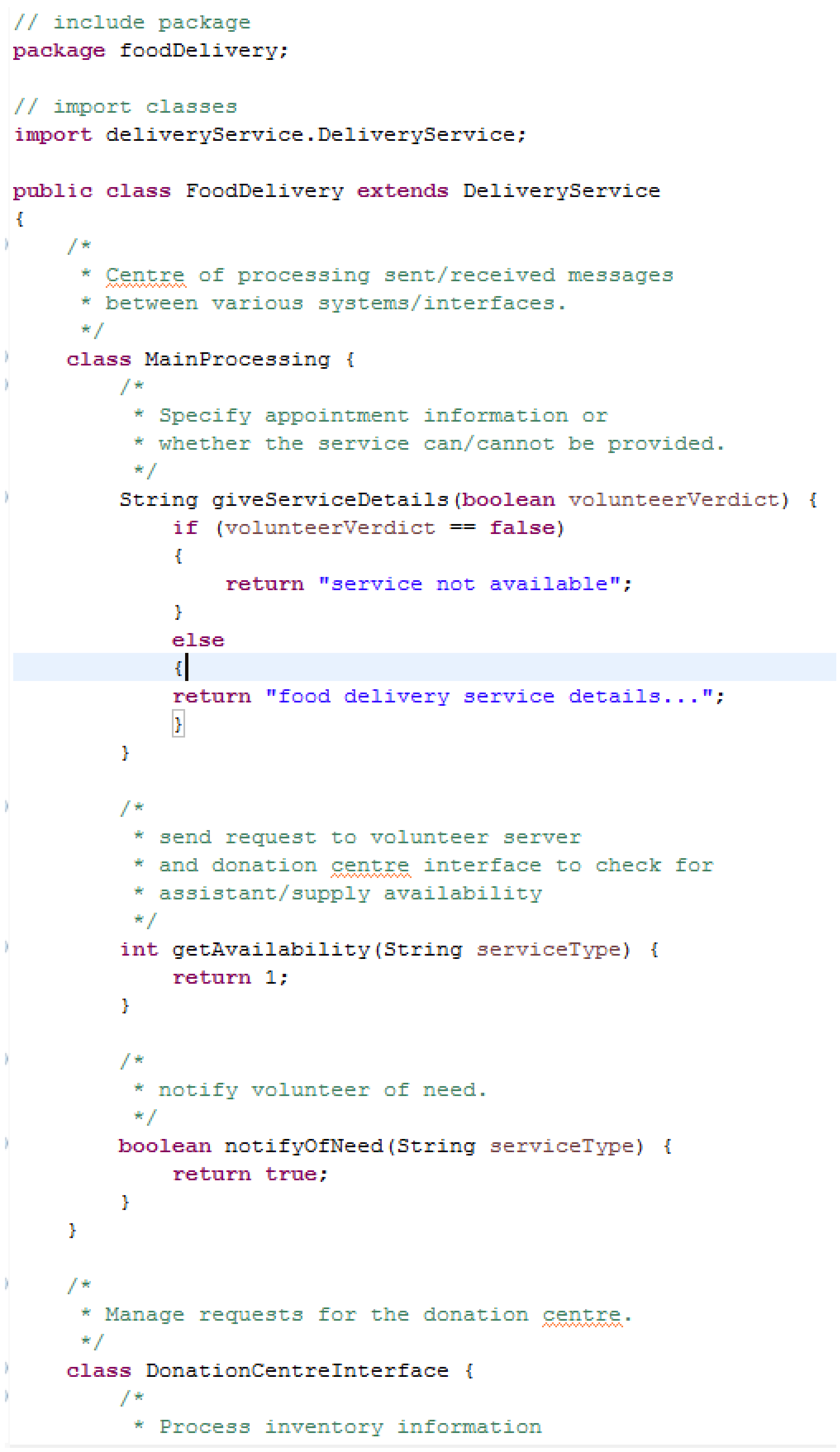
```
    class DonationCentreInterface {
        /*
         * Process inventory information
         * for items sent to the donation centre
         * from the wealth giver.
         */
        boolean inventoryAvailable(String serviceType) {
            return true;
        }
    }
}
```



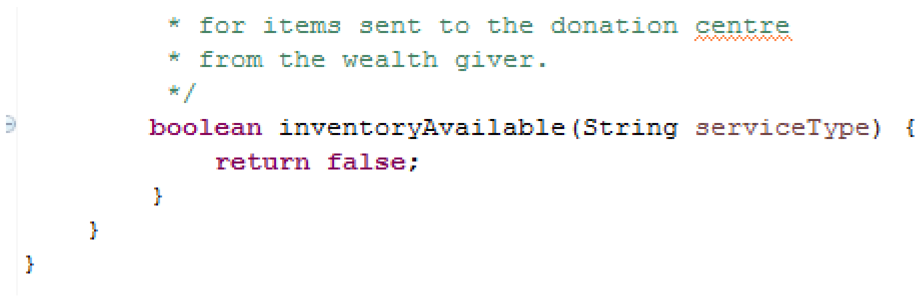
```java
// include package
package financialPlanning;

// import classes
import deliveryService.DeliveryService;

public class FinancialPlanning extends DeliveryService
{
    /*
     * Centre of processing sent/received messages
     * between various systems/interfaces.
     */
    public class MainProcessing {
        boolean volunteerVerdict;
        String serviceType;

        /*
         * Specify appointment information or
         * whether the service can/cannot be provided.
         */
        String giveServiceDetails(boolean volunteerVerdict) {
            if (volunteerVerdict == false)
            {
                return "service not available";
            }
            else
            {
            return "financial planning service details...";
            }
        }

        /*
         * send request to volunteer server
         * and donation centre interface to check for
         * assistant/supply availability
         */
        int getAvailability(String serviceType) {
            return 1;
        }

        /*
         * notify volunteer of need.
         */
        boolean notifyOfNeed(String serviceType) {
            return true;
        }
    }

    /*
     * Manage requests for the donation centre.
     */
```



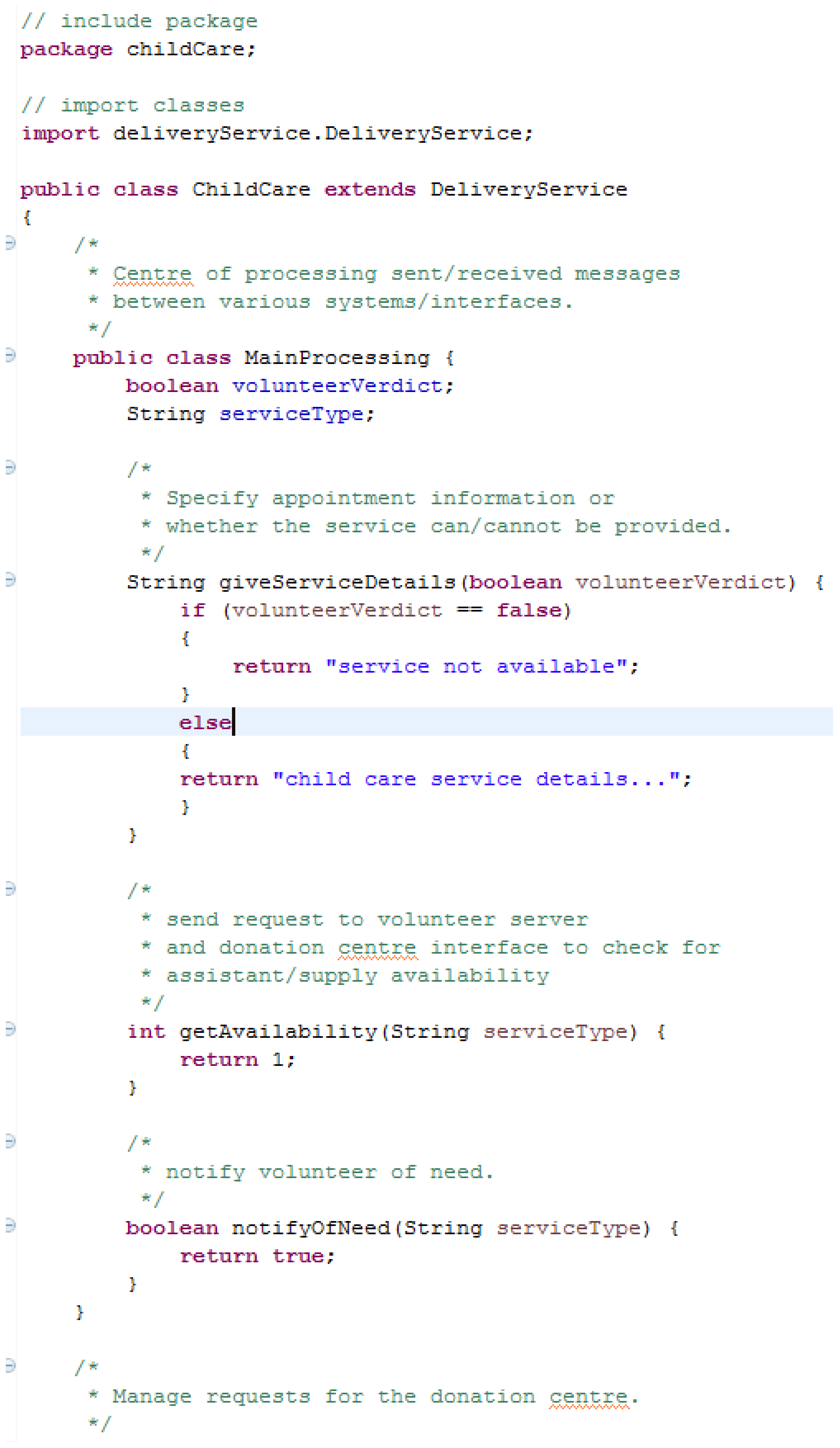
```
    class DonationCentreInterface {
        /*
         * Process inventory information
         * for items sent to the donation centre
         * from the wealth giver.
         */
        boolean inventoryAvailable(String serviceType) {
            return true;
        }
    }
}
```



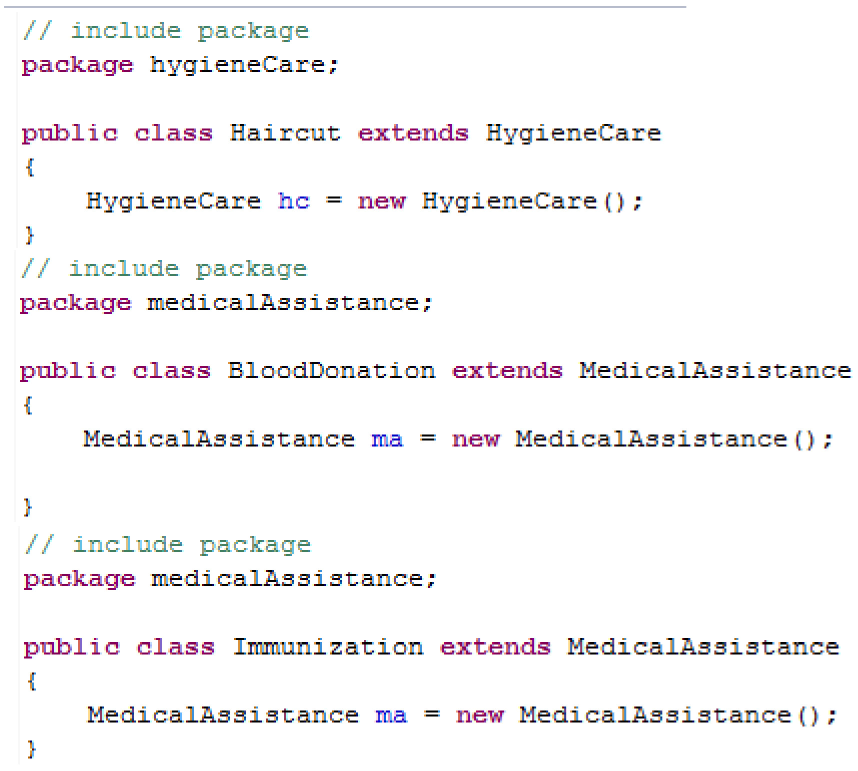
```
// include package
package foodDelivery;

// import classes
import deliveryService.DeliveryService;

public class FoodDelivery extends DeliveryService
{
    /*
     * Centre of processing sent/received messages
     * between various systems/interfaces.
     */
    class MainProcessing {
        /*
         * Specify appointment information or
         * whether the service can/cannot be provided.
         */
        String giveServiceDetails(boolean volunteerVerdict) {
            if (volunteerVerdict == false)
            {
                return "service not available";
            }
            else
            {
            return "food delivery service details...";
            }
        }

        /*
         * send request to volunteer server
         * and donation centre interface to check for
         * assistant/supply availability
         */
        int getAvailability(String serviceType) {
            return 1;
        }

        /*
         * notify volunteer of need.
         */
        boolean notifyOfNeed(String serviceType) {
            return true;
        }
    }

    /*
     * Manage requests for the donation centre.
     */
    class DonationCentreInterface {
        /*
         * Process inventory information
```



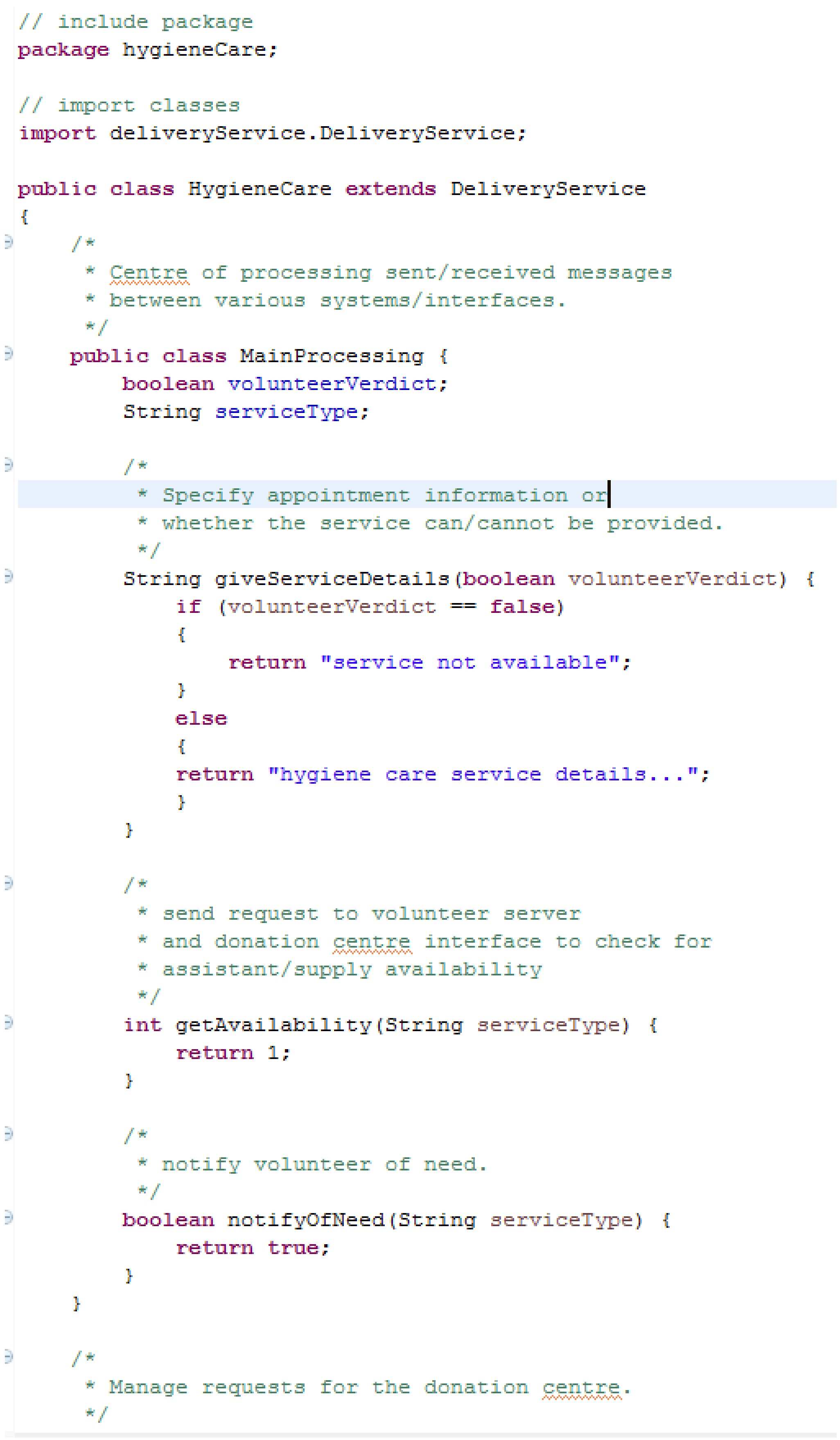
```
         * for items sent to the donation centre
         * from the wealth giver.
         */
        boolean inventoryAvailable(String serviceType) {
            return false;
        }
    }
}
```



```
// include package
package childCare;

// import classes
import deliveryService.DeliveryService;

public class ChildCare extends DeliveryService
{
    /*
     * Centre of processing sent/received messages
     * between various systems/interfaces.
     */
    public class MainProcessing {
        boolean volunteerVerdict;
        String serviceType;

        /*
         * Specify appointment information or
         * whether the service can/cannot be provided.
         */
        String giveServiceDetails(boolean volunteerVerdict) {
            if (volunteerVerdict == false)
            {
                return "service not available";
            }
            else
            {
            return "child care service details...";
            }
        }

        /*
         * send request to volunteer server
         * and donation centre interface to check for
         * assistant/supply availability
         */
        int getAvailability(String serviceType) {
            return 1;
        }

        /*
         * notify volunteer of need.
         */
        boolean notifyOfNeed(String serviceType) {
            return true;
        }
    }

    /*
     * Manage requests for the donation centre.
     */
```



```java
// include package
package hygieneCare;

public class Haircut extends HygieneCare
{
    HygieneCare hc = new HygieneCare();
}
// include package
package medicalAssistance;

public class BloodDonation extends MedicalAssistance
{
    MedicalAssistance ma = new MedicalAssistance();

}
// include package
package medicalAssistance;

public class Immunization extends MedicalAssistance
{
    MedicalAssistance ma = new MedicalAssistance();
}
```



```
// include package
package hygieneCare;

// import classes
import deliveryService.DeliveryService;

public class HygieneCare extends DeliveryService
{
    /*
     * Centre of processing sent/received messages
     * between various systems/interfaces.
     */
    public class MainProcessing {
        boolean volunteerVerdict;
        String serviceType;

        /*
         * Specify appointment information or
         * whether the service can/cannot be provided.
         */
        String giveServiceDetails(boolean volunteerVerdict) {
            if (volunteerVerdict == false)
            {
                return "service not available";
            }
            else
            {
            return "hygiene care service details...";
            }
        }

        /*
         * send request to volunteer server
         * and donation centre interface to check for
         * assistant/supply availability
         */
        int getAvailability(String serviceType) {
            return 1;
        }

        /*
         * notify volunteer of need.
         */
        boolean notifyOfNeed(String serviceType) {
            return true;
        }
    }

    /*
     * Manage requests for the donation centre.
     */
    class DonationCentreInterface {
        /*
         * Process inventory information
         * for items sent to the donation centre
         * from the wealth giver.
         */
        boolean inventoryAvailable(String serviceType) {
            return true;
        }
    }
}
```