2019 25 Nov JC van Staalduinen

M2 System Software Design additional document

Additional information from extended sequence diagrams to Java code

The sequence diagram in the slides assume that the Customer and BookCopy objects were created before the start of the **loan a book** requirement is executed. This is not practical in real life, but seems to be accepted in sequence diagrams. This document extends the sequence and class diagrams for the Loan Books from the library requirement.

Use Case Description for the Library **Loan books** : This Use Case Description is a bit more detailed than the Use Case descriptions you have created up to now. The level of detail in a use case can vary from very high level description (to use in discussions with your user) to a much lower level than this one (to pass on to the programmers).

Librarian	System
1 Select Loan Books	2 Display instruction to scan the pass
3 Scan customer's pass	4 Display information about the customer
	5 Instructs the user to scan the books
6 Scan book barcode	7 Display book information
	8 Registers the loan and displays the loan data
	9 Asks user whether more book need to be scanned
10 User replies	11.1 if "yes" repeat from 5 11.2 if "no" continue with 12
	12 Ask user whether finished
13 User replies	14.1 if "yes" display confirmation 14.2 if "no" display screen where user can select what he/she wants to do next

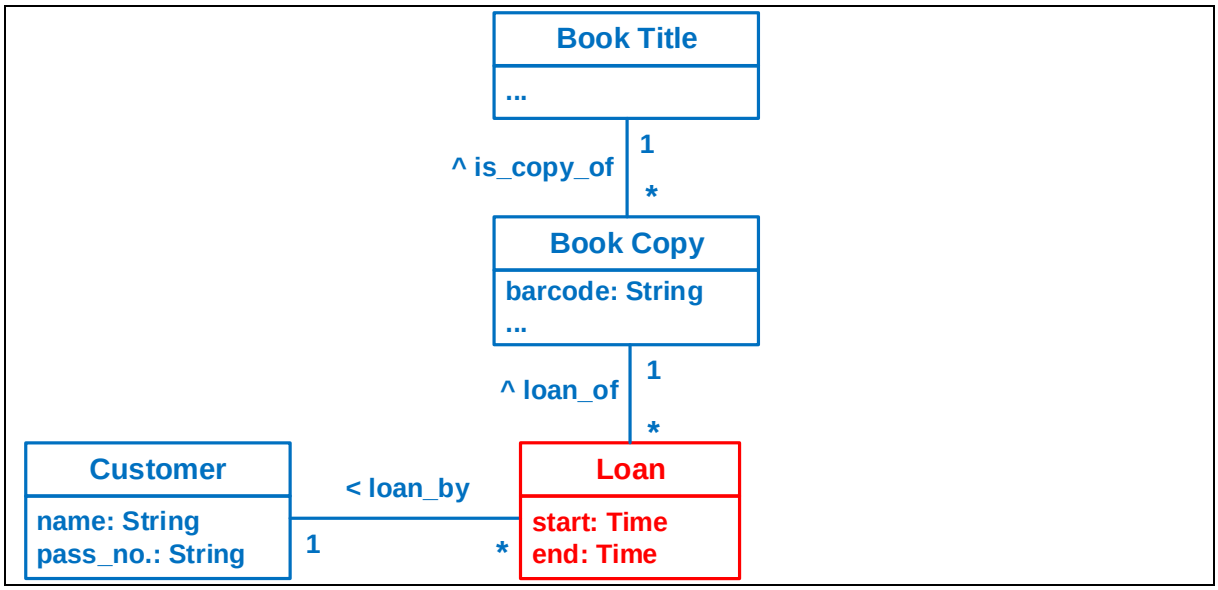
Output of Java program:

Output:

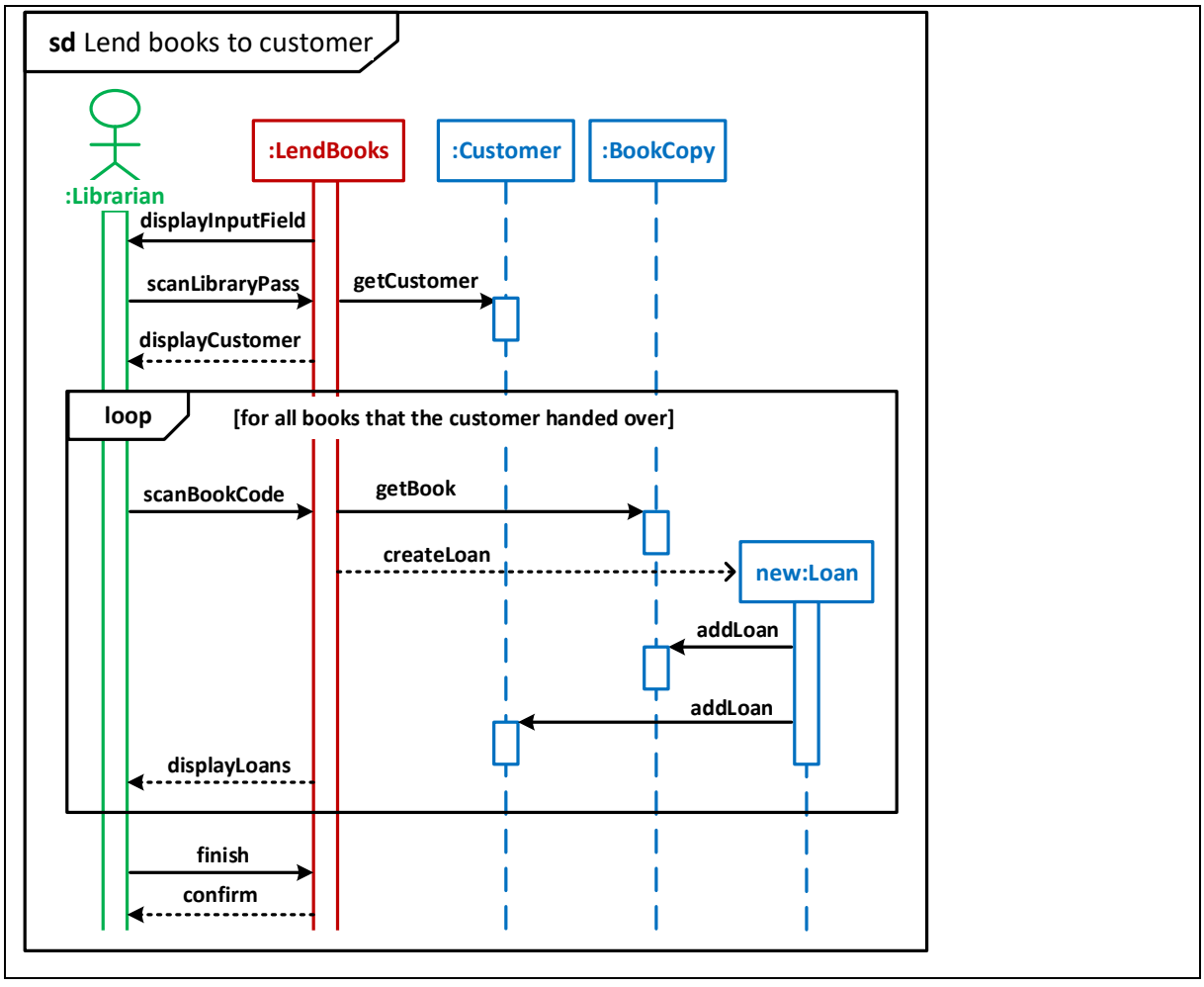
```
Please scan the Customer pass// 2 Display instruction to scan pass
A             // 3 Scan library pass "simulated by keyboard input" customer name
123           //customer ID
A 123 0.0     // 4 Display customer data – name, I and debt
Please enter the book barcode // 5 Instruct user to scan book barcode
234           // 6 Scan book barcode "simulated by keyboard input"
234 How to survive Computer Science at the UT // 7 Book information displayed
//(note this was only a stub so the same book name is returned every time
//in real life a different name will be returned each time
A 123 0.0 234 How to survive Computer Science at the UT 25/11/2019 23/12/2019
//8 loan was created and updated loan information displayed
Do you want to continue? // 9 ask user whether there are more books
Yes           // 10 user replies "yes"
Please enter the book barcode // 11.1 repeat from 5 for second book
345
345 How to survive Computer Science at the UT
A 123 0.0 345 How to survive Computer Science at the UT 25/11/2019 23/12/2019
Do you want to continue?
No            // 10 user replied "no"
Finished?     // 11.2 Continue with 12. Ask whether finished
Yes           // 13 user replied "yes"
```

Goodbye	// 14.1 display confirmation

High level Class Diagram

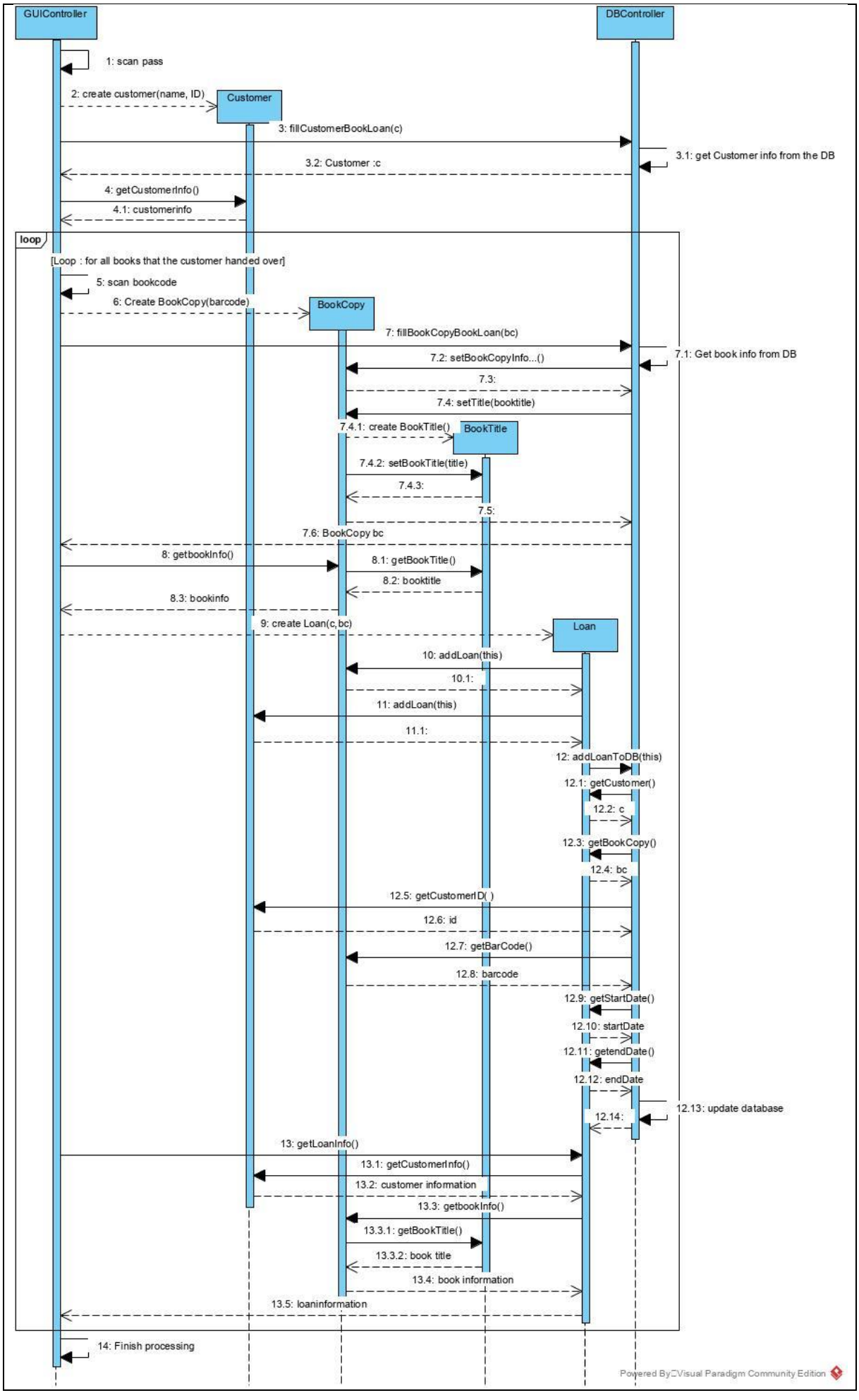


High level Sequence diagram



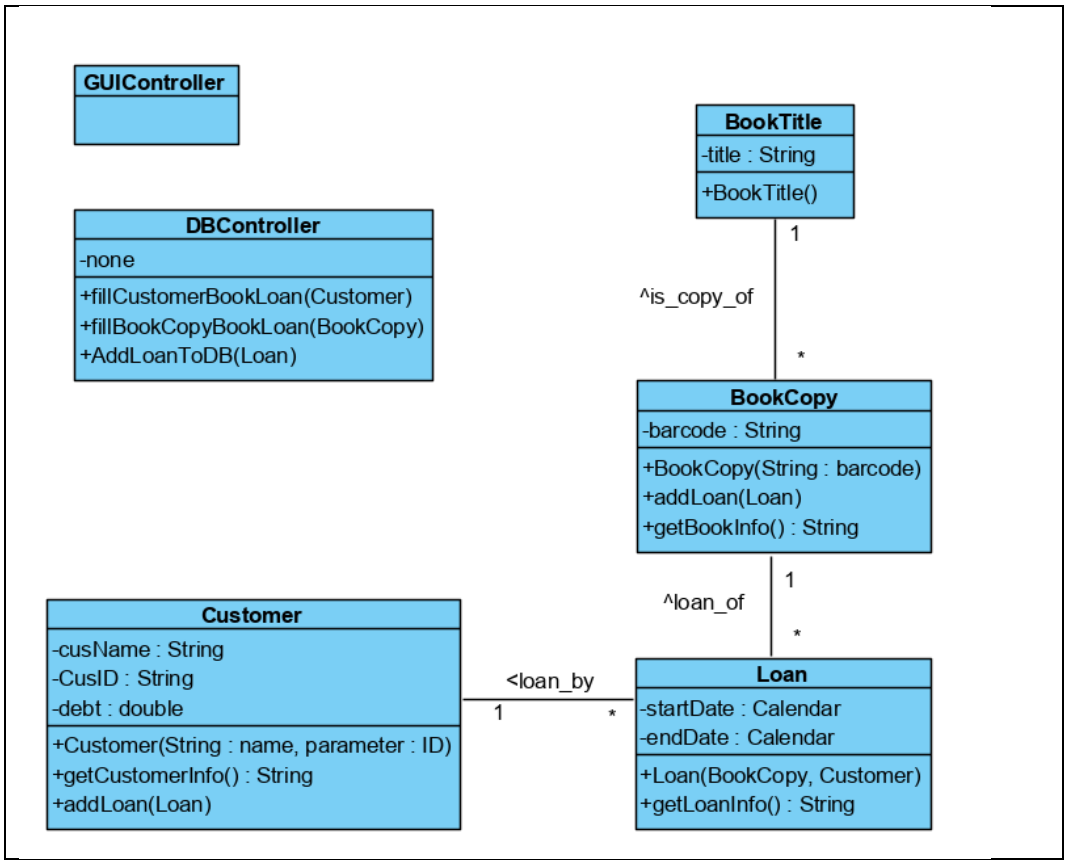
Low level Sequence Diagram

2019 25 Nov JC van Staalduinen



2019 25 Nov JC van Staalduinen

New Class diagram (methods retrieved from the sequence diagram)



Java Code

The controller is combined with the user interface here. If the user uses a GUI the GUI and controller can be separated.

```
import java.util.Scanner;
//import java.util.*;
public class LendBooksController
{

	public static void main(String[] args)
	{
		Scanner scn = new Scanner(System.in);
		String customerName = "";
		String customerID = "";
		Customer c;
		BookCopy bc;
		String continu= "yes";
		String barcode;
		// ArrayList<Loan> ln = new ArrayList<Loan>(); used if you want to
print all loans
		//not used here this time
		Loan loan;

		// displayInputField
		System.out.println("Please scan the Customer pass");

		//scan library Pass
```

```
                customerName = scn.nextLine(); // represents the scanned
                customerID = scn.nextLine();   // library pass

                //create Customer object
                c = new Customer(customerName, customerID);

                //get more customer information from the database
                DBController.FillCustomerBookLoan(c);

                // display customer information
                System.out.println(c.getCustomerInfo());

                while (continu.contentEquals("yes"))
                {
                        // scan book code - we are going to read here
                        System.out.println("Please enter the book barcode ");
                        barcode = scn.nextLine();

                        // create a BookCopy object
                        bc = new BookCopy(barcode);

                        // get more book information from the database
                        DBController.fillBookCopyBookLoan(bc);

                        // display book information
                        System.out.println(bc.getBookInfo());

                        //create Loan
                        loan = new Loan(bc,c);

                        // display loan

                        System.out.println(loan.getLoanInfo());

                        System.out.println("Do you want to continue?");
                        continu = scn.nextLine();
                }

                System.out.println("Finished?");
                continu = scn.nextLine();
                if (continu.contentEquals("yes"))
                {
                        System.out.println("Goodbye");
                }
        }
}
```

```
import java.util.ArrayList;
public class Customer
{
        private String cusName;
        private String cusID;
        private double debt;
        private ArrayList<Loan> booksOnLoan;

        public Customer(String n, String ID)
        {
                setCustomerName(n);
```

```
            setCustomerID(ID);
            booksOnLoan = new ArrayList<Loan>();
      }

      public String getCustomerName()
      {
            return cusName;
      }

      public void setCustomerName(String n)
      {
            cusName = n;
      }

      public String getCustomerID()
      {
            return cusID;
      }

      public void setCustomerID(String ID)
      {
            cusID = ID;
      }

      public double getCustomerDebt()
      {
            return debt;
      }

      public void setCustomerDebt(double d)
      {
            debt = d;
      }

      public ArrayList<Loan> getBooksOnLoan()
      {
            return booksOnLoan;
      }

      public void setBooksOnLoan(ArrayList<Loan> bol)
      {
            booksOnLoan = bol;
      }

      public void addBookLoan(Loan l)
      {
            booksOnLoan.add(l);
      }

      public String getCustomerInfo()
      {
            return cusName + " " + cusID + " " + debt;
      }
}
```

```
import java.util.ArrayList;
public class BookCopy
{
```

```
	private String barcode;
	private BookTitle bookTitle;
	private ArrayList<Loan> booksLoaned ;

	public BookCopy(String b)
	{
		setBarCode(b);
		bookTitle = null;
		booksLoaned = new ArrayList<Loan>();
	}

	public String getBarCode()
	{
		return barcode;
	}

	public void setBarCode(String b)
	{
		barcode = b;
	}

	public void setBookTitle(String title)
	{
		bookTitle = new BookTitle();
		bookTitle.setBookTitle(title);
	}

	// returns the whole booktitle object
	public BookTitle getBookTitle()
	{
		return bookTitle;
	}

	// returns the book's title
	public String getbookTitle()
	{
		return bookTitle.getBookTitle();
	}


	public void setBooksLoaned(ArrayList<Loan> loaned)
	{
		booksLoaned = loaned;
	}

	public ArrayList<Loan> getBooksLoaned()
	{
		return booksLoaned;
	}

	public void addLoan(Loan lo)
	{
		booksLoaned.add(lo);
	}

	public String getBookInfo()
	{
		return barcode + " " + bookTitle.getBookTitle();
	}
```

```
}
```

```
import java.util.ArrayList;
public class BookTitle
{
  private String title; // etc
  private ArrayList <BookCopy> bookcopies; // not used in the
                              // LendBookQuery but is included in the
                              // class as it can be used by other queries
                              // (use cases) such as display all book copies of
this
                              // book title

  public BookTitle()
  {
        setBookTitle("");
        bookcopies= new ArrayList<BookCopy>();
  }

  public void setBookTitle(String t)
  {
        title = t;
  }

  public String getBookTitle()
  {
        return(title);
  }


  public void setBookCopies(ArrayList<BookCopy> bc)
  {
       bookcopies = bc; //Argument:should one do this
                        //because bc points to the original list.
                        //rather make a copy of the ArrayList if there
                        //are reasons to believe that the original array can be
                        //corrupted
  }


  public ArrayList<BookCopy> getBookCopies()
  {
        return bookcopies;  // vary dangerous as the content of the
                            // bookcopies can be altered. Rather make a copy
                            // that is returned
  }

  public void addBookCopy(BookCopy bookc)
  {
        bookcopies.add(bookc);
  }

  //etc

}
```

```
import java.util.*;
```

```java
import java.text.DateFormat;
import java.text.SimpleDateFormat;
public class Loan
{
	private Calendar startDate;  //should be a Time or Date object
	private Calendar endDate;  // should be a Time or Date object
	private BookCopy bookCopy;
	private Customer cust;

	public Loan(BookCopy bc, Customer c)
	{
		setDate();
		setBookCopy(bc);
		setCustomer(c);
		bc.addLoan(this);
		c.addLoan(this);
		DBController.addLoanToDatabase(this);
	}

	public Calendar getStartDate()
	{
		return startDate;
	}

	public Calendar getEndDate()
	{
		return endDate;
	}
	public void setDate()
	{
		startDate = Calendar.getInstance();
		endDate = Calendar.getInstance();
		// endTime = startTime + 30 days
		startDate.setTime(new Date()); // Now use today date.
		endDate.setTime(new Date());
		endDate.add(Calendar.DATE, 28); // Adding 28 days
	}

	public Customer getCustomer()
	{
		return cust;
	}

	public void setCustomer(Customer c)
	{
		cust = c;
	}

	public BookCopy getBookCopy()
	{
		return bookCopy;
	}

	public void setBookCopy(BookCopy bc)
	{
		bookCopy = bc;
	}

	public String getLoanInfo()
```

```
	{
		DateFormat dateFormat = new SimpleDateFormat("dd/MM/yyyy");

		return cust.getCustomerInfo() + " " + bookCopy.getBookInfo() + " "
					+ dateFormat.format(startDate.getTime())
					+ " " + dateFormat.format(endDate.getTime());
	}
}
```

```
import java.util.*;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
public class DBController
{
	public static void FillCustomerBookLoan(Customer c)
	{
		// code to open database plus SQL to get the information
		// for this customer from the database, using c.getCustomerName()
		// and c.getCustomerID() to get the information from the
		// customer object to include in the SQL to
		// search the database for the specific customer
		// The return information from the search will include the customer
		// debt. Here we are actually using a stub to create a debt of 0
		c.setCustomerDebt(0.0);
	}

	public static void fillBookCopyBookLoan(BookCopy bc)
	{
		String booktitle;
		// as with customer, the information of the book copy
		// is searched in the DB using SQL. We can get info about the
		// damage to the book, etc etc and set this in the BookCopy
		// object (for simplicity not included here).
		// E.g. bc.setBookQuality(...); bc.setBookVersion(...); etc.
		// We can also find the BookTitle information with the same SQL.
		// We are going to set booktitle in BookCopy. Assume the book title
		// we retrieved from the database is "How to survive
		//  Computer Science at the UT"

		booktitle = "How to survive Computer Science at the UT";
		bc.setBookTitle(booktitle);
	}

	public static void addLoanToDatabase(Loan l)
	{
		Customer c = l.getCustomer();
		BookCopy bc = l.getBookCopy();
		String customerID = c.getCustomerID();
		String barcode = bc.getBarCode();
		Calendar start = l.getStartDate();
		Calendar end = l.getEndDate();
		//customerID, barcode, start and end are used by the SQL
		// create a load record and, if necessary update the
		// specific customer and book copy records
	}
}
```