CSC 308 Midterm Exam

Instructions: *The exam is open-book, open-note. It is worth a total 120 points. Use back of pages or extra paper as necessary. You have both the lecture and lab times to complete the exam, but it's likely you will not need this much time.*

1. Software Process Questions

a. (4 points) Describe one important difference between a team that develops custom software compared to a team that develops off-the-shelf software.

A custom software team has actual end users from the organization for which the custom software is being developed. In an off-the-shelf software team, there are representative end users, rather than the specific people who will ultimately purchase the product.

Other acceptable answers can make reference to the following kinds of people being on off-the-shelf teams: marketers, focus group participants; and the following kinds of people being on custom teams: owning managers, surveyed end users.

b. (4 points) Are there any fundamental incompatibilities between agile software development and formal methods of software development? If there are any incompatibilities, state one. If there are none, say so.

None.

A plausible argument can be made for an incompatibility stemming from the lack of a complete model being developed in an agile process, thereby precluding the complete formal specification of the model. An answer along these lines receives full credit.

c. (4 points) Describe what you believe to be the most significant problem with the original "waterfall" process for software development.

Water sometimes flows "uphill". I.e., it is unrealistic to have no iteration at all in a software process; the original waterfall process was presented as iteration-free.

-- the following answer is also fully acceptable --

Testing is shown as the very last step of the process, rather than having testing shown as a pervasive step of the process.

2. Requirements Analysis Questions

In the current Calendar Tool requirements, the people who serve as system administrators are anonymous. This is because the official definition of "Calendar Tool Administrator" is anyone who happens to know the password for the Calendar Tool Administration program. No record is kept of who these people are.

The Calendar Tool customer wants to change this aspect of the requirements. Specifically, there needs to be a managed list of people who serve as Calendar Tool administrators

The the following information is to be kept for each registered administrator:

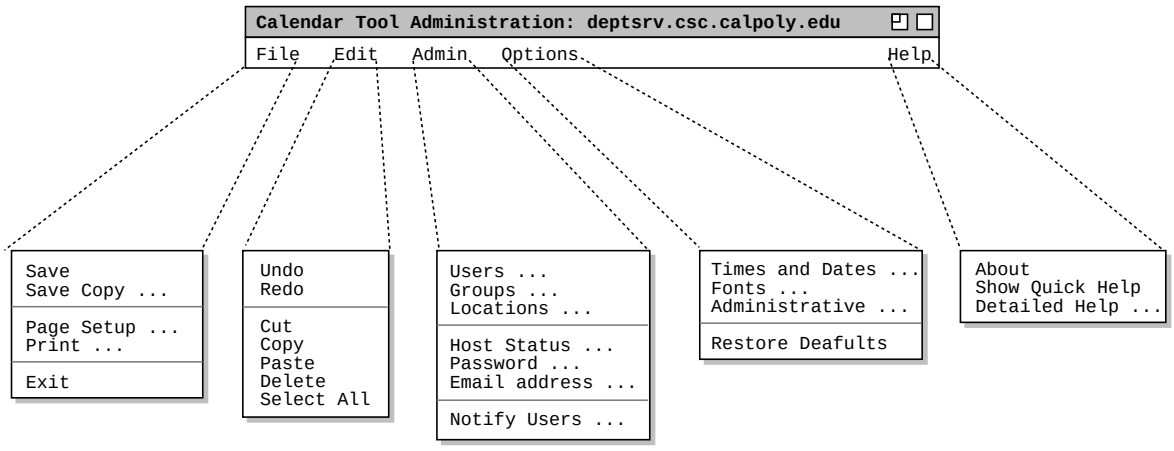
- Full name (first, last, middle)
- Password
- Phone number (area, number)
- Email address

There is one pre-defined administrator, with a pre-defined password. From there, additional administrators can be added, changed, and deleted.

The list of administrators needs to be distinct from regular Calendar Tool users. It's fine if someone wants to be both an administrator and a regular user. In such a case, the person will have two separate Calendar Tool records -- one as admin, and another as regular user.

Given this background information, answer the following questions.

a. (16 points) How does the above new functionality fit into the current UI of the Calendar Tool Administration program? Specifically, what changes need to be made to the following menu-based UI? For clarity, the `Admin` menu is described in the paragraph immediately following the picture:



The top three commands in the'`Admin`' menu provide access to the three databases of: Calendar Tool users, user groups, and meeting locations. Administrative users can add, change, and delete database information. The '`Host Status`' command allows the administrator to view and edit information about the Calendar Tool central host computer. The '`Password`' command is used to set the administrative access password. '`Email Address`' is used to set the email address through which regular users contact the administrative staff. '`Notify Users ...`' provides the means to send a message to some or all Calendar Tool users.

Give your answer to Question 2a by drawing a picture of where your new command(s) go(es) in the menus, and what other UI changes you think need to be made. Do not redraw the entire set of menus, but only as much as necessary to provide adequate context for your answer. Also, you can refer as necessary to your answer to Question 2b, to avoid redrawing UI screens. Note that the command(s) you add should be consistent in style with the existing command menus.

I'd put a new "Admins ..." command on the Admin menu, like this:

```
Admin
-------------------
Users ...
Groups ...
Locations ...
-------------------
Host Status ...
Admins ...
Email Address ...
-------------------
Notify Users ...
```

The 'Password ...' item in the Admin menu is replaced with an 'Admins ...' item. The old Password item is no longer needed, since each admin now has her or his own password. As shown in the answer to question 2b, pressing 'Admins ...' brings up an editing dialog.

b. (42 points) Write a scenario that describes how an administrative user adds a new administrator to the system. Provide no more than three screen shots, which you consider to be the three most important screens to convey these requirements. Supply necessary supporting narrative.

When the user selects the 'Admins ...' item in the 'Admin' menu, the system displays the window shown in Figure 1

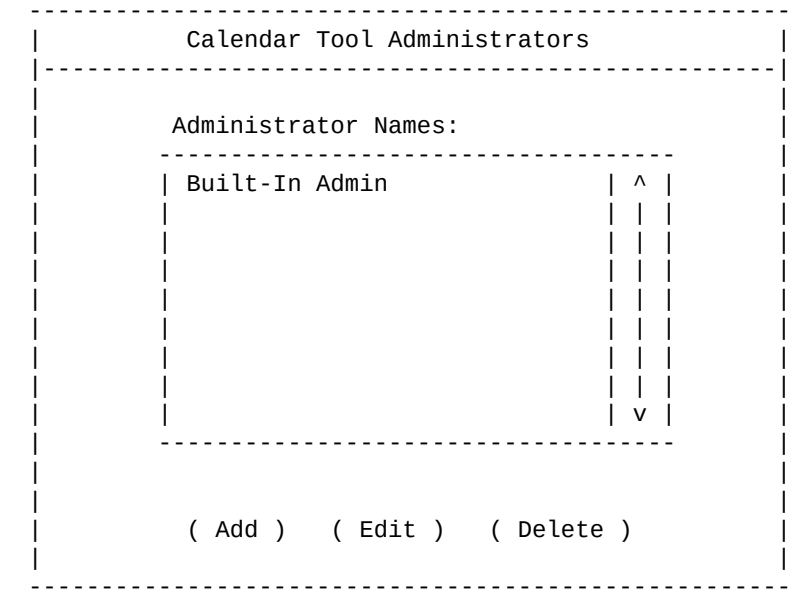


Figure 1: Dialog to manage Calendar Tool administrators.

The window in Figure 1 has a scrolling list of administrator names. There is one pre-defined administrator named "Built-In Admin". The buttons at the bottom allow the user to add, edit, and delete administrators.

When the user presses the 'Add' button in Figure 1, the system displays the dialog shown in Figure 2.

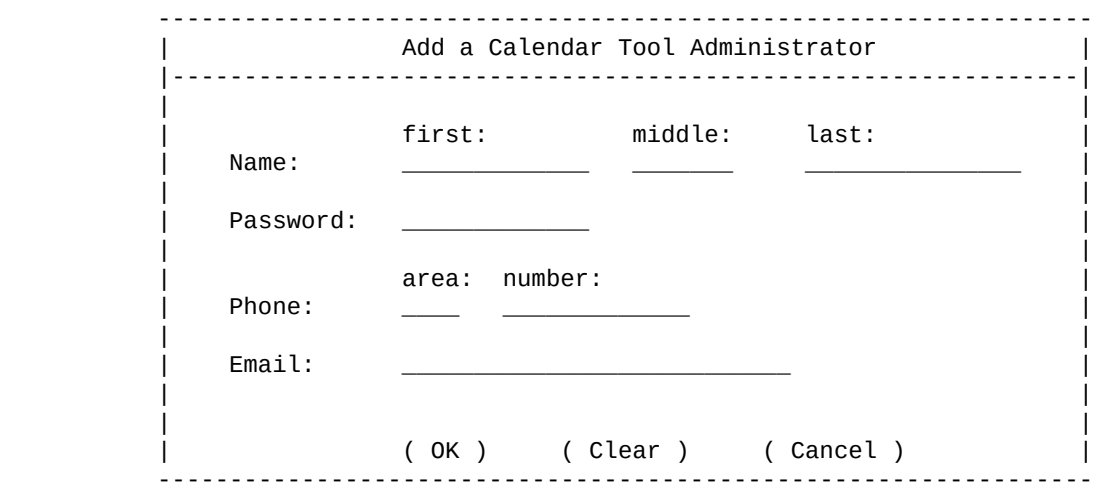


Figure 2: Dialog to add a system administrator.

The three components of the name are free-form strings. The name must be unique, i.e., no two or more administrators can have exactly the same name, where the comparison is based on the concatenation of first, middle, and last. The password and email address are free-form strings. The phone area and number must be three and seven digits, respectively.

Figure 3 shows the user having filled in information for a typical administrator.

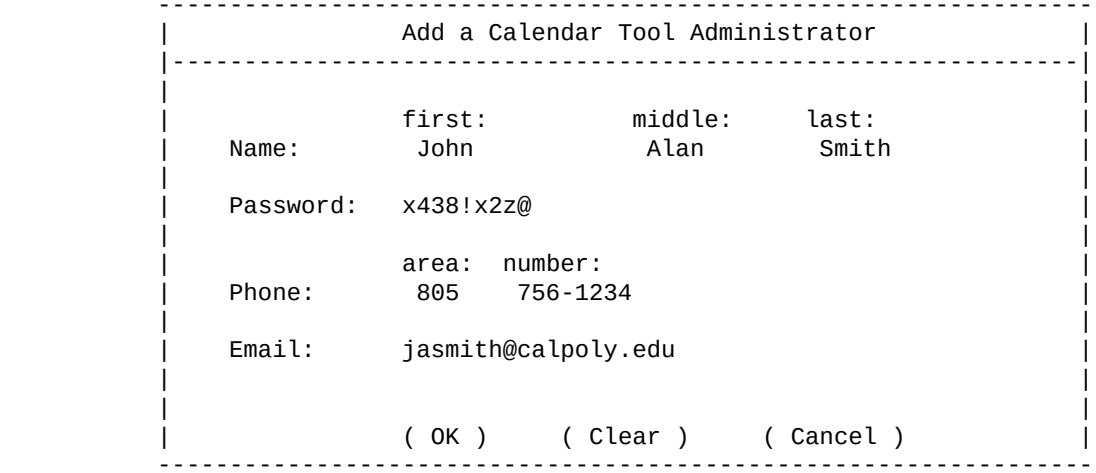


Figure 3: Filled-in information for an administrator.

When the user presses OK, the system adds the new administrator to the current administrator list.

Question 2 Grading Notes:

Part a:

- *Full credit for a reasonable placement of 'Admin ...' item.*
- *Up to -5 points for excessive redraw of menus*
- *Up to -5 points for any particularly inconsistent command structure, such as unexplained Add/Edit/Change sub-menus instead of "..." style dialog launch.*

Part b:

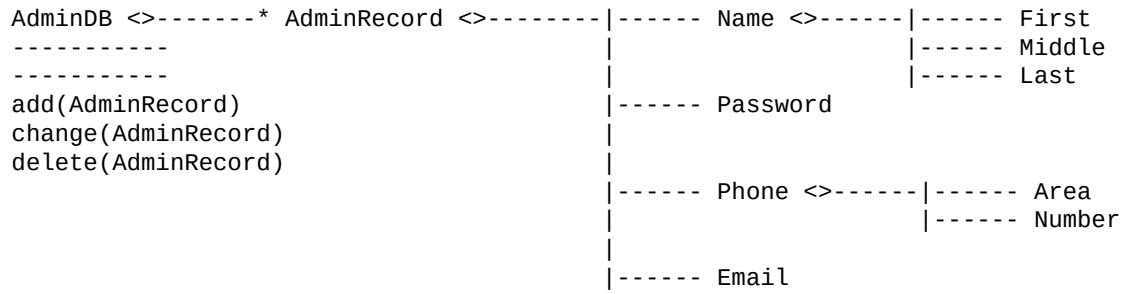
- *14 points for each of three action/response use cases*
  - *7 points per screen*
  - *7 points per accompanying narrative*
- *General deductions:*
  - *-7 pts if empty and filled-in input dialogs not shown in some form*
  - *-3 pts for each prose style violation, up to -9*
  - *-3 pts for each egregious misplacement of figures, up to -6*
  - *-2 pts for each missing significant UI component, in particular title bar and text field labels*

3. Modeling Questions

a. (20 points) Based on your answer to Question 2, derive Java objects for the collection of administrator records, subsidiary objects, and associated operations. The model should be compilable.

```
import java.util.Collection;

abstract class AdminDB {
    Collection<AdminRecord> data;
    abstract void add(AdminRecord ar);
    abstract void change(AdminRecord old_ar, AdminRecord new_ar);
    abstract void delete(AdminRecord ar);
}

abstract class AdminRecord {
    Name name;
    String password;
    Phone phone;
    String email;
}

abstract class Name {
    String first;
    String middle;
    String last;
}

abstract class Phone {
    int area;
    int number;
}
```

b. (10 points) Draw a UML diagram that is equivalent to the preceding Java definitions.



c. (20 points) Using Spest notation, define preconditions and/or postconditions that formally specify the following requirements:

- all registered administrators have unique names, i.e., no two or more registered administrators can have the all of the same first, middle, and last names
- an administrator password must contain at least one digit and at least one letter; the Java library functions `Character.isDigit(char ch)` and `Character.isLetter(char ch)` are used to determine whether a character is a digit or letter

It's up to you to determine which method or methods these conditions are specified for, and whether they are preconditions, postconditions, or both. Once you have made these determinations, write your answer by giving the full signature of the method(s), preceded by the appropriate Spest specification.

```
/*
    pre:
        // There's no existing record with same name as input record.
        !exists (AdminRecord ar_other;
              data.contains(ar_other);
                  ar.name.first.equals(ar_other.name.first) &&
                  ar.name.middle.equals(ar_other.name.middle) &&
                  ar.name.last.equals(ar_other.name.last))

                      &&

         // Input password has at least one digit.
         exists (int i; i>=0 && i<ar.password.length();
             Character.isDigit(ar.password.charAt(i)))

                      &&

         // Input password has at least one letter.
         exists (int i; i>=0 && i<ar.password.length();
             Character.isLetter(ar.password.charAt(i)));
*/
abstract void add(AdminRecord ar);
```

Question 3 Grading Notes:

Part a:

- *1 pt for import*
- *10 pts for AdminDB (2 for header, 2 for data field, 2 each for method)*
- *9 pts for AdminRecord (either all-in-one or separate-classes is acceptable)*
- *-2 pts for each significant inconsistency with the scenarios, up to -6; notable inconsistencies are missing components and substantial deviations in object names compared to the scenarios*

Part b:

- *same as part a, with point values halved*
- *-2 pts for each inconsistency with Java definition, up to -6.*

Part c:

- *10 pts for first precondition clause*
- *5 pts each for second and third precondition clauses*