Session 1: Menu Programs

At the end of this session participants will be able to:

- Understand how to design the screens of a menu program
- Use variables of type pff to launch one CSPro application from another
- Create a simple menu program to launch the main data entry program
- Pass data from the menu program to the data entry program

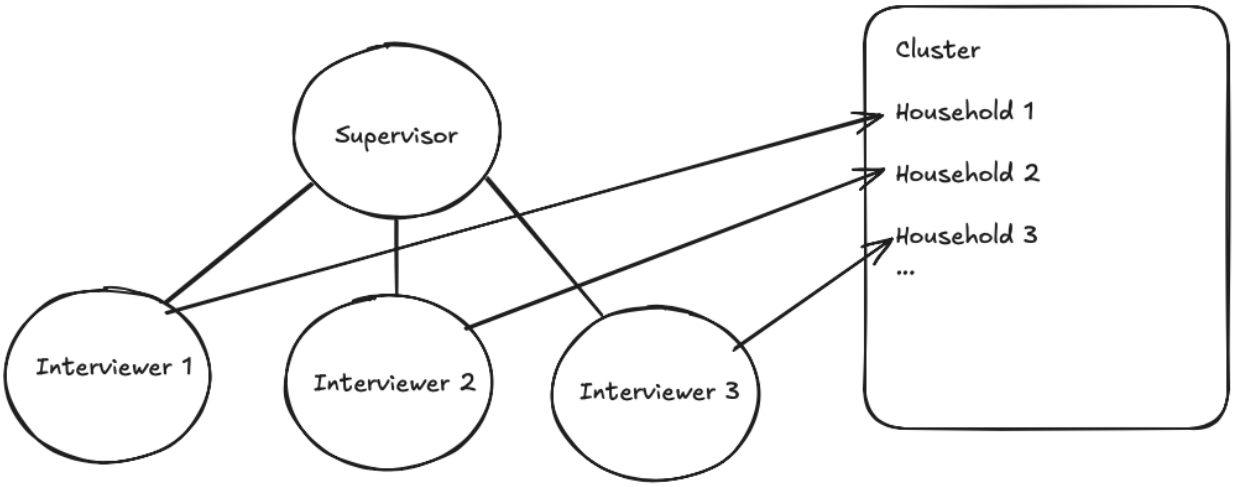
What is a Menu Program?

A menu program is a CSPro data entry application that is used to manage the data entry workflow. A menu program is not used for capturing any interview data itself. Instead, it launches other data entry programs for interviews. Menu programs generally have some or all of the following functions:

- Launches other CSPro applications to do data collection (often pre-filling id items)
- Show reports on progress, summary statistics
- Manage user access through usernames/passwords
- Manage household listing and interview assignments
- Data synchronization

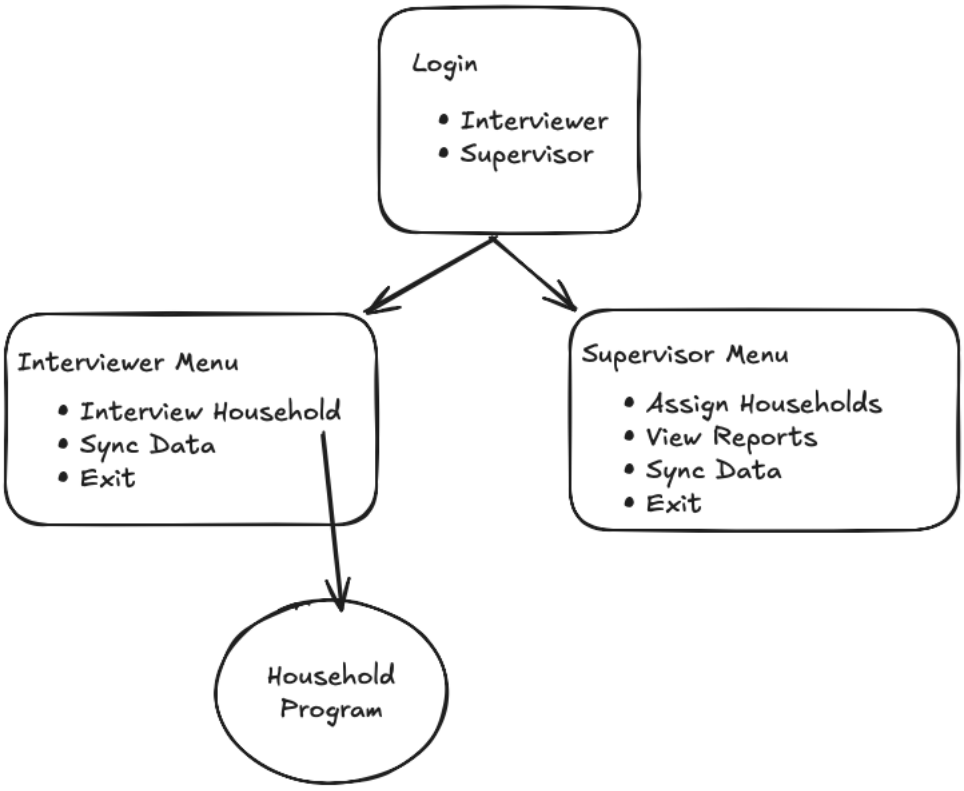
The Example Survey

As an example, we will a simple household survey for the fictitious country of Popstan. This is a sample survey that collects basic demographic and economic data on individuals in households. The sample of households is divided into clusters. The field staff consists of teams of three interviewers and a supervisor. Each cluster is assigned to a supervisor who is responsible for assigning individual households within the cluster to interviewers and verifying their work.



Designing the Screens and Flow

Once you decide on which functions you want your menu program to perform, the next step is to design the screens and how they link together. This is most easily done as a diagram like the one below for a fairly simple menu program.



Creating the Dictionary and Forms

Once you have the screens, the next step is to create a new CSPro application for your menu program. Let's name ours "*Menu"* and put it in the folder Popstan*HouseholdSurvey/Menu*. When creating the application, choose "Entry" as the category and "Operational Control Application" as the type. This creates a CSPro data entry application with default settings that are standard for menu applications but fundamentally it is no different from any other data entry application in CSPro.

Each menu screen will be a different variable in the dictionary. The value set for the variable will show the available menu choices to the enumerator on that screen. In the menu example above, we will have three variables:

- LOGIN (value set: Interviewer - 1, Supervisor - 2)
- INTERVIEWER_MAIN_MENU (value set: Interview Household - 1, Sync Data: 2, Exit - 9)
- SUPERVISOR_MAIN_MENU (value set: Assign Households – 1, View Reports – 2, Sync Data - 3, Exit - 9).

We will keep the default id-item that CSPro creates for us. The menu program dictionary doesn't really need an id-item since we do not need to save our menu choices to the data file. However, CSPro requires that we have at least one id-item so we will keep it.

Create the items in the dictionary and then create a form and drop the items onto the form. Do not drop the id-item onto the form. We can leave it off the form since we will never actually write out a completed case from the menu program. Unlike typical data entry applications, menu programs tend not to have a linear flow. As a result, the order of the variables on the form is less important. We will use skips and reenters to move from one menu to another.

Menu Program Logic

The logic for processing the menu choice for each screen goes in the postproc of the variable for the menu. For example, to process the login menu field in our example we would have the following logic:

```
PROC LOGIN

// Go to the appropriate menu for the role chosen
if $ = 1 then
    skip to INTERVIEWER_MAIN_MENU;
else
    skip to SUPERVISOR_MAIN_MENU;
endif;
```

If the user selects to login as an interviewer, we skip to the interviewer main menu field to show the interviewer menu, otherwise we skip to the supervisor main menu field to show the supervisor menu.

Handling the interviewer menu is similar. Here we will create a user defined functions to launch the household program and to sync data.

```
PROC INTERVIEWER_MAIN_MENU
postproc

// Handle the menu choice
if $ = 1 then
    // Household questionnaire
    launchHouseholdDataEntry();
elseif $ = 2 then
    // Sync data
    synchronizeData();
elseif $ = 9 then
     // Exit
     stop(1);
endif;

// Show interviewer menu again
reenter;
```

It is important to make sure that after the postproc of the menu field we do not let CSEntry continue to the next field, otherwise after the interviewer launches the household application, they would end up in the next field, which, in this case is the supervisor menu. To prevent this, we put a reenter at the end of the postproc so that we go back into the same menu field again.

It looks kind of strange that when we go back into a menu, the previous choice is still selected. We can prevent this by clearing the choice in the onfocus of the field.

```
onfocus
// Clear previous choice
$ = notappl;
```

The supervisor menu is similar to the interviewer menu:

```
PROC SUPERVISOR_MAIN_MENU
onfocus
// Clear previous choice
$ = notappl;

postproc
// Handle the menu choice
if $ = 1 then
    // Assign households
elseif if $ = 2 then
    // Show reports
elseif $ = 3 then
    // Sync data
    synchronizeData();
elseif $ = 9 then
    // Exit
    stop(1);
endif;

// Show supervisor menu again
reenter;
```

Launching one CSPro Application from Another

Let's fill in the function to launch the household data entry program. We can launch other CSPro programs from within our data entry program using a variable of type `pff`. A `pff` variable represents a pff file. You can load the contents of an existing pff file into a `pff` variable using the load() function and passing it the path to the file. You can then run the associated CSPro application by calling the `exec()` function. This will start the application using the parameters in the pff file.

The following logic will launch a data entry application using the pff file *Household.pff* in the sibling directory to the menu program directory.

```
function launchHouseholdDataEntry()
    pff householdPff;
    householdPff.load("../Household/Household.pff");
    householdPff.exec();
end;
```

Making the Entry Program Return to the Menu Program

When we exit the household application, we want to return to the menu, but the menu program stopped when we launched the household application. To get it to restart we add a parameter to the pff file to tell it to restart the menu program when it exits.

```
function launchHouseholdDataEntry()
    pff householdPff;
    householdPff.load("../Household/Household.pff");
    householdPff.setProperty("OnExit", "../Menu/Menu.pff");
    householdPff.exec();
end;
```

Tips for Menu Programs

Now that we have the menu program to launch the Household program, we don't want the Household application to appear in the application list on the mobile device. By default, CSPro creates the application list from all the pff files in the CSEntry directory on the device. To prevent an application from being listed, you can edit the pff file in the pff editor and choose "Hidden by Default" or "Never" for the "Show in Application Listing" option.

Saving Login when Returning to the Menu

It is annoying to have choose the role every time you come back into the menu program from the household application. To avoid this, we can save the login using a persistent variable so that it will keep its value when the application is reloaded.

```
PROC GLOBAL

// Save the login between runs of the application
persistent numeric savedLogin;

PROC LOGIN

preproc
// Check for a saved login and skip the question if there is one
if savedLogin > 0 then
    noinput;
endif;

postproc
// Save the login for next time
savedLogin = LOGIN;

// Go to the appropriate menu for the role chosen
if $ = 1 then
    skip to INTERVIEWER_MAIN_MENU;
else
    skip to SUPERVISOR_MAIN_MENU;
endif;
```

We also need to clear the persistent variable when we logout.

```
PROC INTERVIEWER_MAIN_MENU
onfocus
// Clear previous choice
$ = notappl;

postproc
// Handle the menu choice
if $ = 1 then
    // Household questionnaire
    launchHouseholdDataEntry();
elseif $ = 2 then
    // Sync data
    synchronizeData();
elseif $ = 9 then
     // Exit
     savedLogin = 0; // Clear saved login
     stop(1);
endif;
```

```
// Show interviewer menu again
reenter;
```

Passing Data to the Household Application through the PFF

The household questionnaire has a question for the name of the interviewer (A6). Rather than have the user fill in that question for each household, they can enter it once in the menu program, and we can fill it in automatically in the household questionnaire. First let's have the user enter their name on the login screen in addition to their role so we capture it when they login.

The parameters section of the pff file allows you to pass any value to your data entry program. We will use this to pass the name of the interviewer in the menu program to the household questionnaire program.

```
function launchHouseholdDataEntry()
    pff householdPff;
    householdPff.load("../Household/Household.pff");
    householdPff.setProperty("OnExit", "../Menu/Menu.pff");
    householdPff.setProperty("INTERVIEWER_NAME", USER_NAME);
    householdPff.exec();
end;
```

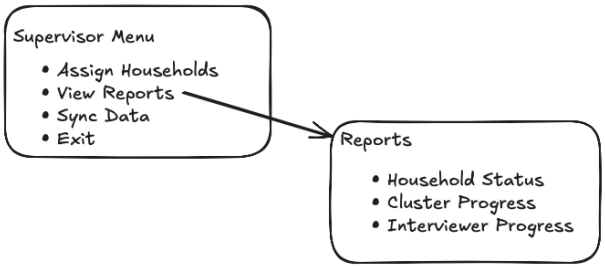
The next step is to modify the household application to pre-populate the interviewer name using the pff file parameter. This can be done using the sysparm() command which retrieves a parameter by the name from the pff. We can do this in the INTERVIEWER _NAME preproc in the household questionnaire application. The argument to sysparm () must EXACTLY match the string passed as the first parameter to pff.setProperty() in the menu program. The sysparm () function always returns its result as a string and returns an empty string if no matching parameter was set in the pff object in the menu program.

```
PROC INTERVIEWER_NAME
preproc

string interviewerName = sysparm("INTERVIEWER_NAME");
if interviewerName <> "" then
    $ = interviewerName;
endif;
```

Exercises

1. Implement the menu option for *View Reports* in the supervisor menu. It should show an additional menu with the following three options: 1) Household Status, 2) Cluster Progress, 3) Interviewer Progress. Do not implement the reports themselves. To make sure the menu works you can just show a message when the user picks a report saying which report was picked (for example – "User picked Household Status").



2. In addition to the name of the interviewer, the household questionnaire asks for the interviewer code (A7). Collect the interviewer code in the menu program when the user logs in. Pass the interviewer code from the menu program to the household program and fill it in automatically in the household program just like we did for the interviewer's name.
3. Use persistent variables to save the interviewer name and interviewer code in the menu application so that the interviewer doesn't have to enter them every time they log in. Make sure to clear them when the user logs out.