DSB001: Skills Bootcamp in Software Engineering, Day 18, Lab

The HTTLAP process

Step 1 Understand the problem

- Reformulate, divide, find the unknowns
- Turn requirements into **tests**

Step 2 Devise a plan

- **Data structures** and **algorithms** (narrative, diagrams, pseudo-code, etc.)
- Revisit the **tests**

Step 3 Implement the plan (**prototype**: paper, software, etc.)

Step 4 Check the result - run the **tests**

Step 5 Lessons learnt (**agile**: loop back to 3)

Step 6 Document the solution (such that others can understand and **reuse** it)

Though written with a Java implementation in mind, you may choose to complete this lab task in any language covered in the module.

Specification - Part One

Model a network of typical office equipment (a printer, a photocopier, a scanner, a coffee machine and a vending machine) along with a simple monitoring/management tool which will be able to notify users of the status (online, offline or in error state) of each connected piece of equipment. All machines have unique alphanumeric codes and can process jobs (only one at a time, no tracking of when jobs were entered into the system is needed at this time). A job will have a code (alphanumeric and unique), an owner (the person who ordered it) and a description (e.g., “Print document xyz.” or “Make a large latte macchiato.”). The job and machine codes always start with three letters that describe the type of the job/machine, followed by a combination of three digits. The letters at the start of the job code are:

- PRT print job / printer
- SCN scan job / scanner
- CPY copy job / copier
- CFE make coffee job / coffee machine
- VND dispense food from vending machine job / vending machine

All machines go online and offline in the same way, by setting / resetting the `online` flag (boolean field). Similarly, all machines can reset, thus exiting their error state - the `error` flag becomes `false`. All machines accept jobs in the same way, namely if there is no other current job, if there is no error state, if the machine is online and if the first three letters in the job code match those in the machine code. If at least one of the first three conditions is not met, the state of the machine remains unchanged. If the fourth condition is broken, the machine goes in an error state.

Each machine processes a job in different way.

- Printers display the description of their job in the console and then set the `job` field to `null`.
- Copiers accept jobs with extra data indicating the number of copies to make for that job. The copier will prefix each copy it produces with the count of each copy along with the total number requested in that job (e.g. for a job requesting three copies, the first copy output will be prefixed with 1 of 3, the second with 2 of 3 etc.). Finally, the copier will set the `job` field to `null`.
- Scanners modify their job to get it ready to be passed on to a printer. To that effect, scanners swap the job's owner with their machine code, replace the first three letters in the job code with `PRT` and **do not** set the `job` to `null` afterwards. They do, however, have a method to allow clearing the job explicitly.
- Coffee and vending machines check the job's owner. If it starts with 1, the owner is senior management and gets coffee / food for free. Otherwise, the owner is notified via a message displayed in the console that a standard amount (10p) has been taken out of his/her account as payment for the job. The `job` is set to `null` afterwards.

The manager is **not** a piece of office equipment. It is a fully digital system running off a server that monitors all machines in the office and has the main task of supervising the job assignment process. To this end, the manager will store a list of all the jobs currently active (that have not yet been processed) and a record of all available machines. The manager performs the following tasks:

- Assigns jobs. Every job in the list will be assigned to the first machine found that is capable of performing the job (the first three letters of the code match those of the job). If no compatible machine is found, the appropriate message is displayed in the console.
- Processes jobs. Every machine in the system is asked to process the job previously assigned to it. If the machine is a scanner, the manager will explicitly retrieve the job processed by the scanner and assign it to the first printer available. At the end of the process, all jobs get deleted from the manager's memory - even the ones that were not assigned to machines.

Specification - Part Two

Every piece of office equipment:

- Registers a listener. The machine receives an object representing the listener and stores it internally (assigns it to a field declared for this purpose).
- Notifies the listener. Whenever the machine goes online, offline or enters an error state, it prompts the internal listener to show its full description including the new status.

The manager is the only machine that can act as a listener. Once it receives a notification from a machine in the system, it displays the full description of that machine.

Registering and notifying listeners are not specific to office equipment only (the air conditioning unit - modelled in a separate hierarchy - implements those two operations as well). Similarly, the manager is not the only one that can show the status of a machine: other managers, produced by other organisations, should be usable in the system. Other monitoring systems exist that are able to perform that task. You are **not** asked to model air conditioning units or other monitoring systems, but, the fact that they exist and may be added to your model by someone else at a later point should give you an indication of where to put methods `registerListener()`, `notifyListener()` and `showStatus()`.

Specification - Part Three

An exception should be generated if

- A machine changes status and there is no listener on record. The exception message should include the id of the issuing machine.
- A food processing machine is set to work at a temperature outside its pre-defined safety limits. The exception message should include the illegal temperature and the safety limits (lower and upper).

Apply the HTTLAP Process to Part One of the Spec

Follow the steps in the beginning of the lab sheet to implement the first part of the specification. Make sure to draw a class diagram and update it as you progress through the HTTLAP steps: this will greatly help you, and others, understand the structure of your system.

Apply the HTTLAP Process to Part Two of the Spec

Revisit your notes on interfaces. Adapt your design, and then your implementation, in order to include this new functionality. Ensure that the system is fully tested.

Apply the HTTLAP Process to Part Three of the Spec

Revisit your notes on exceptions. Adapt your design, and then your implementation, in order to include this new functionality. Ensure that the system is fully tested.

What to Submit

Do not skip the pseudo-code writing steps. Submit the final version of the class diagram (complete with classes, fields and methods) and the code archive.