CptS 322 Software Engineering Principles I

Homework 5

1. We have studied four architectural styles: data-centered, data-flow, call-return, and layered. Now give an example software project for which you think each of these styles would fit the best. Briefly justify the choice of architectural style for the example project you chose. [12 points]

Data-centered architecture

A speech system: there is a central database hosting a large number of sample data; then different clients can be independently working around the database (speech sampling, speech recognition, speech-text conversion, etc.) [3points]

Data-flow architecture

A university payroll system: start with the input of raw income data, go through different levels of processing (e.g., department to add bonus, school to deduct donations, college to subtract budget-cut share, health service to deduct medical cost, so and so forth, and finally human resource to compute tax), finally produce the paycheck. [3 points]

Call-return architecture

A restaurant assistant system (as exemplified in class): components dealing with various tasks (order processing, bill computation, etc.) connected with different levels of control (top-level executive/main, preprocessing management, output control, etc.). [3 points]

Layered architecture

A typical modern operating system: hardware layer at bottom, then drivers and kernel, followed by third-party libraries, OS utilities and finally built-in application software. [3 points]

2. There are two measures of functional independence in software design: cohesion and coupling. Explain in your words what each of these measures

means. Is it better to have higher/lower cohesion? Why? Is it better to have higher/lower coupling? Why? [18 points]

Cohesion measures the degree to which different pieces of a module/component are related or connected together to perform a focused function. [3points]

Coupling measures the degree to which a module/component is connected to or depends on other modules/components in the software system. [3points]

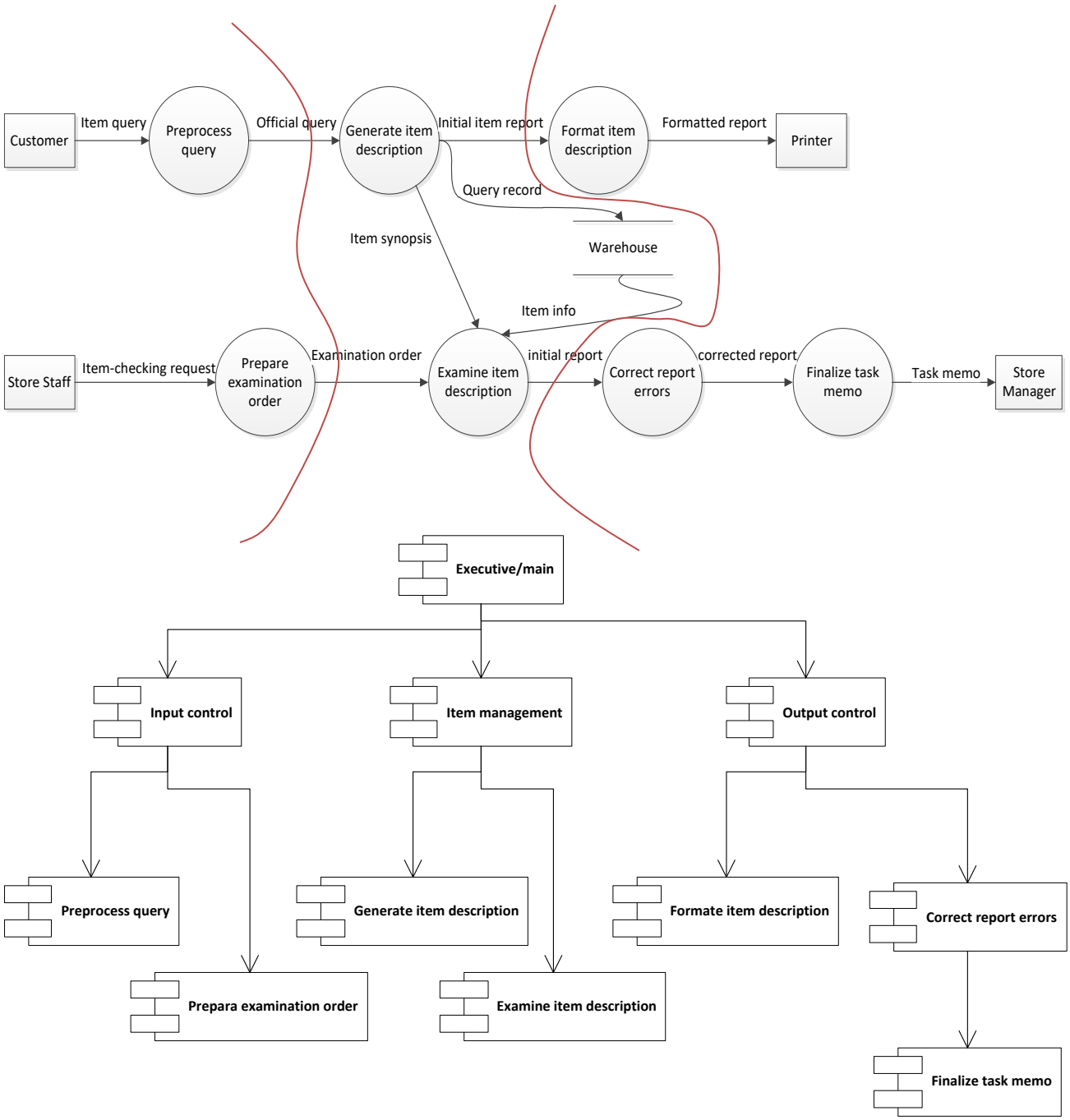
It is better to have high cohesion [3points]
because [3points]

- A high-cohesion module/component is simpler since all its parts focus on one thing.
- A high-cohesion module/component is more reusable because developers will find the component/module they need more easily given the cohesive set of operations provided by the module/component.

It is better to have low coupling [3points]
because with low coupling [3points]

- A change in one module/component usually requires fewer changes in other modules/components.
- There will be less inter-module/component dependence making it easier to test each.

3. The following data flow diagram is part of the flow models for the requirements of ACE hardware store management software. It describes the process for responding to a customer request for querying about an item. Derive call-return architecture from it for this software, and draw the component diagrams that represent your architectural design. [20 points]



*IT is okay to have different partitioning of the DFD thence different architectural design (for example, 'correct report errors' could be part of the output processing hence become a component underneath "output control".

Notes:

-Submit a single PDF for these questions to Canvas.

-Use standard UML notations when drawing the diagrams (no hand-drawing is accepted).