Software Engineering (Adv CS II, 320212)

Final Exam Spring 2012

Logistics

- You have 90 minutes (sharp) for the test.
- You can reach 54.5 points if you solve all problems. You will only need 52 points for a perfect score, i.e. 2.5 points are bonus points.
- Mark all sheets you deliver with your name to make sure it can get graded. We cannot grade if not present or illegible!
- *Different problems test different skills and knowledge, so do not get stuck on one problem.*

Name:

(To be used for correcting, do not write into box below)

Task	1.1	1.2	1.3	2.1	2.2	2.3	3.1	**Total**
Total	3	3	2	3	2	4	2	
Reached								

Task	3.2	4.1	4.2	5.1	5.2	5.3	5.4	5.5	6.1	6.2	7.1	7.2	7.3	**Total**
Total	3	4	2	5	2	3	3	4	4	2	1	2	2	**54.5**
Reached														

1 User Interface Design

Task 1.1 (3p): List and explain the main analysis techniques available during the design of computer user interfaces.
Answer: Task Analysis, Interviewing and questionaires, Ethnography and a short explanation for each
Task 1.2 (6p): Explain Pressman's Golden Rules and give, for each one, an example where it is fulfilled and where it is violated; explain exactly why it fulfils/violates the rule.
Answer:

- Place user in control – In a word processor, allow the user to undo changes, hide internal implementation(e.g. the html code used to generate the document)
- Reduce user's memory load – Use a disk icon for save operation, use shortcuts consistently across the app
- Make interface consistent – Keep the same design over different pages in your website. Keep the search form in the same place on each page, etc

Task 1.3 (2p): Your company is requested to design a menu-based interface for a walk-up system (e.g. an ATM). What factors should you consider when designing the interface?
Answer:

- Used by a variety of people with different technical background => interface should be simple and it should be easy for the user to correct mistakes.
- Take into account users with disabilities => do not base your interface on color recognition, design large buttons etc.

2 Graphical User Interface Technology

Task 2.1 (3p): Describe the three main architectural concepts underlying modern GUI systems. Note: this is not MVC!
Answer: Event-driven programming, Widgets, Interactor-tree + brief description for each

Task 2.2 (2p): Explain the MVC pattern.
Answer: Model–View–Controller (MVC) is a design pattern for computer user interfaces that divides an application into three areas of responsibility:

- Model: the domain objects or data structures that represent the application's state.
- View, which observes the state and generates output to the users.
- Controller, which translates user input into operations on the model.

Task 2.3 (4p): Your company has been asked to provide a GUI for the campus library. Design a simple system (by way of model and views you would implement) based on the following specifications:

- Users can see if a book is available
- Users can book conference rooms
- Staff can add descriptions to books

Answer (possible):

Models: Book, Resource, User, Role etc
Views:

- Book Availability: displays a list of reservations on the selected book
- Resource Booking: displays the status of a certain resource(e.g Room 145)
- Staff Area: Displays a list of books in stock where staff can edit descriptions

Controller:

- Book Available: Uses the User and Book model to display Book Availability view
- Resources: Uses the Resource and User model to display the Resource Booking view
- Admin Area – Uses the Staff Area view with the User and Book model

Etc.

3 Web Content Management Systems

Task 3.1 (2p): Describe 4 attributes which differentiate Web applications from desktop applications.
Answer: Network intensiveness, Concurrency, Performance, Availability + description

Task 3.2 (3p): Sketch a possible WCMS architecture with configured business logic. What are the benefits of this implementation and of WCMSs in general?
Answer: Higher flexibilility for defining new business rules, non-developers can add new rules to the system etc.
Consistency, Personalization, Better access and workflows etc
Diagram

4 Language Processing

Task 4.1 (2+2 pts): Write a bison grammar for a calculator that uses postfix notation.
Example inputs:

1 2 + <EOF> (corresponding infix notation: 1 + 2)
3 4 + 5 *<EOF> (corresponding infix notation: (3 + 4) * 5)

The calculator understands integer numbers and the four arithmetic operations of plus ('+') and minus ('-') and multiplication (‘*’) and division (‘/’). Any number of whitespaces is allwed between the tokens. When encountering end-of-file on the input stream, the overall operation result is returned by the parsing routine, *yyparse()*.
Hints:

- You do not need to provide a complete bison source, just the grammar rule section including the associated action code (in particular: no token definitions, no *main*(), no error handling, no (f)lex scanner code). Assume a token NUM representing a number.
- Syntax is not graded per se, however, the concepts must come across clearly and the use of bison and C/C++ must be adequate.

Answer:
%token NUM
%%

```
input: exp    { $$ = $1; }
    ;
exp
    : NUM                { $$ = $1;              }
    | exp exp '+'        { $$ = $1 + $2;         }
    | exp exp '-'        { $$ = $1 - $2;         }
    | exp exp '*'        { $$ = $1 * $2;         }
    | exp exp '/'        { $$ = $1 / $2;         }
    | exp exp '^'        { $$ = pow ($1, $2); }
    | exp 'n'            { $$ = -$1;             } /* Unary negation */
    ;
%%
```

Task 4.2 (2p) Write (f)lex regular expressions which print “WORD” whenever a word is encountered (i.e., a sequence of alphanumeric characters starting with an alphabetical character) and print out “NUMBER” whenever an (unsigned integer) number is encountered. Separating whitespace is ignored.

Hints:

Syntax is not graded per se, however, the concepts must come across clearly and the use of (f)lex and C/C++ must be adequate.

Answer:

```
[0123456789]+              printf("NUMBER\n");
[a-zA-Z][a-zA-Z0-9]*       printf("WORD\n");
[ \n\t]                    ;
```

Task 4.3 (3p): Explain how a C compiler generates executable code from the input source code and libraries. In doing so, name the resp. components performing a particular subtask.

5. Software Process and Project Management

Task 5.1 (1.5p): What are milestones, tasks and deliverables in the context of project activity organization?

Answer: Milestones are end-point of a process activity, where a formal report of progress is presented to management.

Tasks organize the work to be done and the responsibilities.

Deliverables are the project results delivered to the customers.

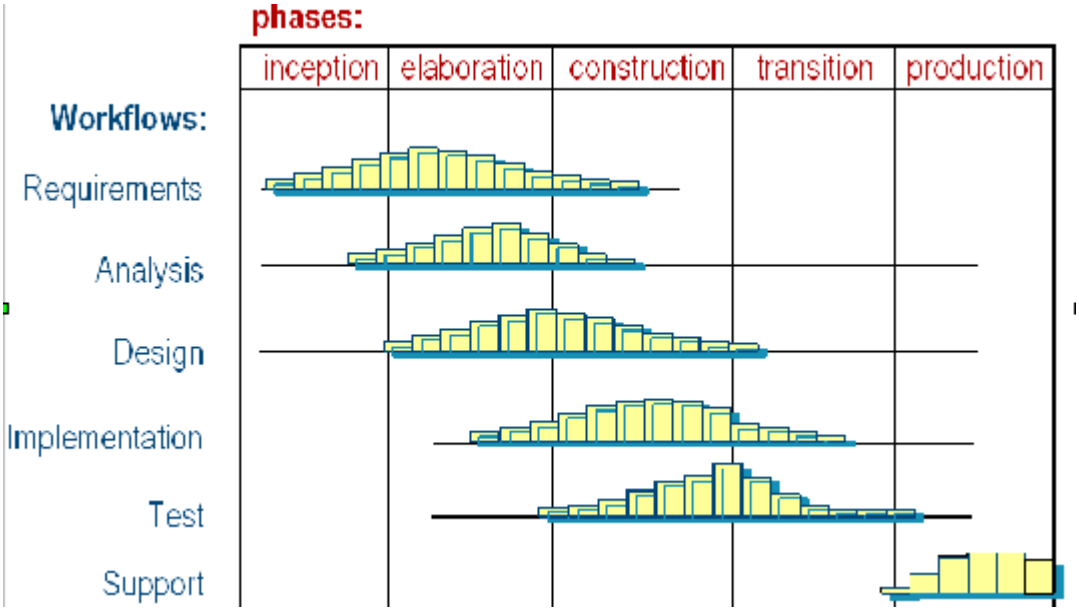
Task 5.2 (2p): The table below presents a number of activities, duration and dependencies. Draw a PERT chart showing the project schedule.

Task	**Duration (days)**	**Dependencies**
T1	10	
T2	20	T1, T3

T3	5	
T4	5	T2
T5	15	T7
T6	20	
T7	15	T3
T8	30	T5, T2, T6

Answer: activity chart

Task 5.3 (3p): Below is the process model adopted by the Rational Unified Process (RUP). Explain how this two-dimensional view of the software development process encompasses/ generalizes most other process models. Sketch the matrix for the iterative development approach.



Task 5.4(3p): What is extreme programming? Discuss one advantage and one disadvantage.
Answer:
A variation of iterative development based on very small increments. New versions may be built several times a day, increments are delivered to the customer every ~ 2 weeks, all tests must be run for every build.
Advantages: XP delivers cheaper software, compared to other models, which is likely to do do exactly the customer wants. It also creates working software faster, and that software tends to have very few defects.
Disadvantages: It's difficult to get many developers to accept the practices, and it takes a lot of discipline to keep doing them all. Customers may not like the idea of having to be so involved.

Task 5.5 (2p + 2p): Two software engineers are having trouble choosing the best process model for their projects and they decide to ask for your advice. The first one needs to build an operating system suitable for controlling nuclear reactors, while the second one is engaged in a startup company doing innovative software. What would you advise them and why?
Answer:
nuclear reactor os => waterfall, big system, stability needed. Startup: agile, many changes are going to follow, startup implies testing the market reaction and continuously updating the application and changing the functionality.

6. CMMI

Task 6.1 (1p): What is Capability Maturity Model Integration (CMMI)? .
Answer: CMMI is a model used to assess the capabilities of organizations to manage their IT processes.
Goals:

- Corrective actions are managed to closure when the project's performance or results deviate significantly from the plan. (PA: Project Monitoring and Control);
- Actual performance and progress of the project is monitored against the project plan (PA: Project Monitoring and Control);
- The requirements are analyzed and validated and a definition of the required functionality is developed. (PA: requirements development);
- Root causes of defects and other problems are systematically determined. (PA: causal analysis and resolution);
- The process is institutionalized as a defined process. (PA: generic goal).

Task 6.2 (1p): Explain one aspect where CMMI is superior to CMM.

Task 6.3 (1p): What is the difference between staged and continuous CMMI?
Answer: In the staged CMMI model, each maturity level has process areas and goals while the continuous model is finer-grained: it considers individual or groups of practices, assesses their use and assigns one maturity level per area.

7. Software Ethics

In the short story "Pragma Suppress" a technical failure causes several lives.
Task 7.1 (1p): What was the technical reason?
Answer: Array access out of bounds because the real-world measurements didn't correspond with the ones declared in the software.

Task 7.2 (2p): What was the organizational reason? (Hint: it was not a single person's failure, but a combination of adverse effects).
Answer: Corporation's decision to buy a cheaper machine that lead to the "PRAGMA supress" command, Lindner's failure of notifying the software company of the changes made to the area, the software company for allowing memory allocation and array access without correct

compiler instructions

Task 7.3 (2p): In class another – related – example of a catastrophic failure was presented which has happened actually on June 4, 1996; what went wrong there? (Hint: it is enough to focus on the software aspects).
Answer: The crash of the Arianne rocket due to an integer overflow problem. A conversion from 64 to 16 bits was not protected which lead to an overflow.