NTNU
Norwegian University of Science and Technology

Faculty of Informatics, Mathematics and Electronics

Department of Computer and Information Sciences

ENGLISH

Examination results will be announced: 28. June

Exam in the subject
TDT4240 Software Architecture

Wednesday 7. June 2006
9:00 am – 1:00 pm

Aids code C:

Simple calculator allowed.
These specified printed documents are allowed:

- IEEE (2000), "IEEE Recommended Practice for Architectural Description of Software-Intensive Systems", Software Engineering Standards Committee of the IEEE Computer Society.
- Kruchten, P. (1995), "The 4+1 View Model of Architecture", IEEE Software, 12(6).
- English-Norwegian dictionary (or to your native language if your not Norwegian) and/or a English thesaurus (English-English).

Contact person during the exam:

Associate professor Alf Inge Wang, phone 73594485, mobile phone: 92289577

The points show how much each problem is worth in this exam. For each problem, each question has the same weight unless otherwise stated. The exam has 5 problems giving a total of 70 points. The remaining 30 points are credits awarded from the robot project.

Good Luck!

Magne Syrstad and Finn Olav Bjørnson

Controlled 29th of May 2006

Problem 1: Various questions (10 points)

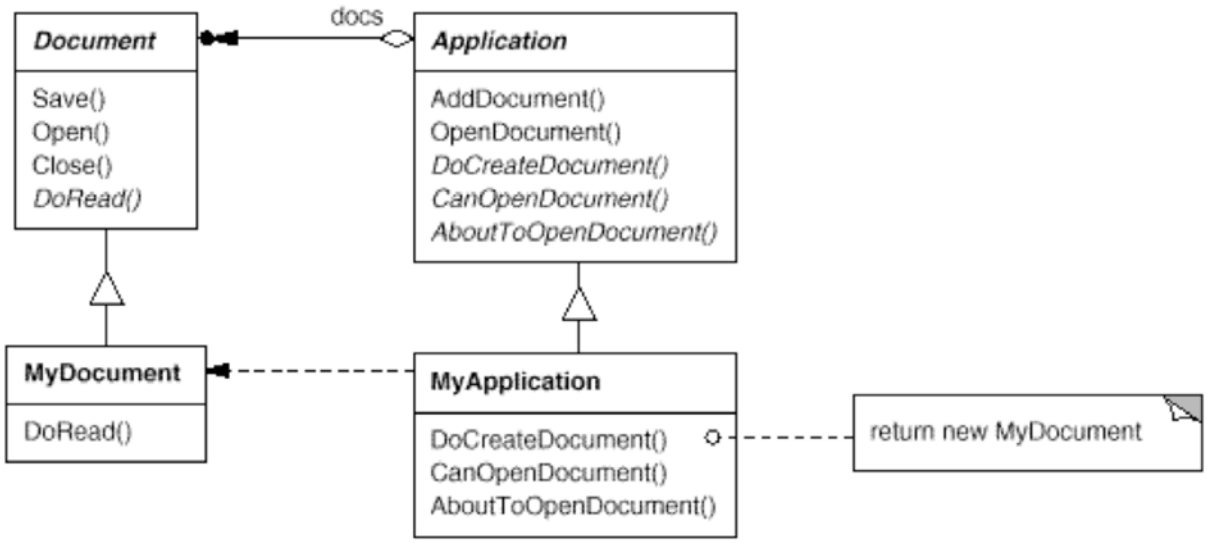
Answer these questions short:

1.1 What is Bass, Clements and Kazman's definition of Software Architecture (the definition in the textbook)?
1.2 How can information be extracted when you are reconstructing a software architecture?
1.3 What is architectural drift?
1.4 What is an architectural pattern and give one example of an architectural pattern?
1.5 What of the following quality area(s) is/are related to scalability: Availability, modifiability, performance, security, testability, and/or usability.
1.6 Why is an enterprise architecture important for big companies like Telenor?
1.7 Where does the architectural design fit into the waterfall process model for software development?
1.8 Why is it useful to describe an architecture through different viewpoints and views?
1.9 How can interface mismatch of using Off-The-Shelves components in a software architecture be repaired?
1.10 How should you design your architecture to support variation points in a product line?

Problem 2: Design patterns (10 points)

2.1 Describe design pattern (5 points)

What is the name of the design pattern shown below and what is the purpose of this design pattern?



2.2 Description of design patterns (5 points)

Name five items (parts) that should be in a description of a design pattern and describe why these items (parts) are important in a design pattern description.

Problem 3: Quality Scenarios (10 points)

Create one quality scenario for availability and *one quality scenario for performance* for the system described below. The scenarios should be fully specified according to the textbook, you can describe the scenario as a diagram or as a table, and the scenarios should be useful and understandable for the stakeholder *user* (player).

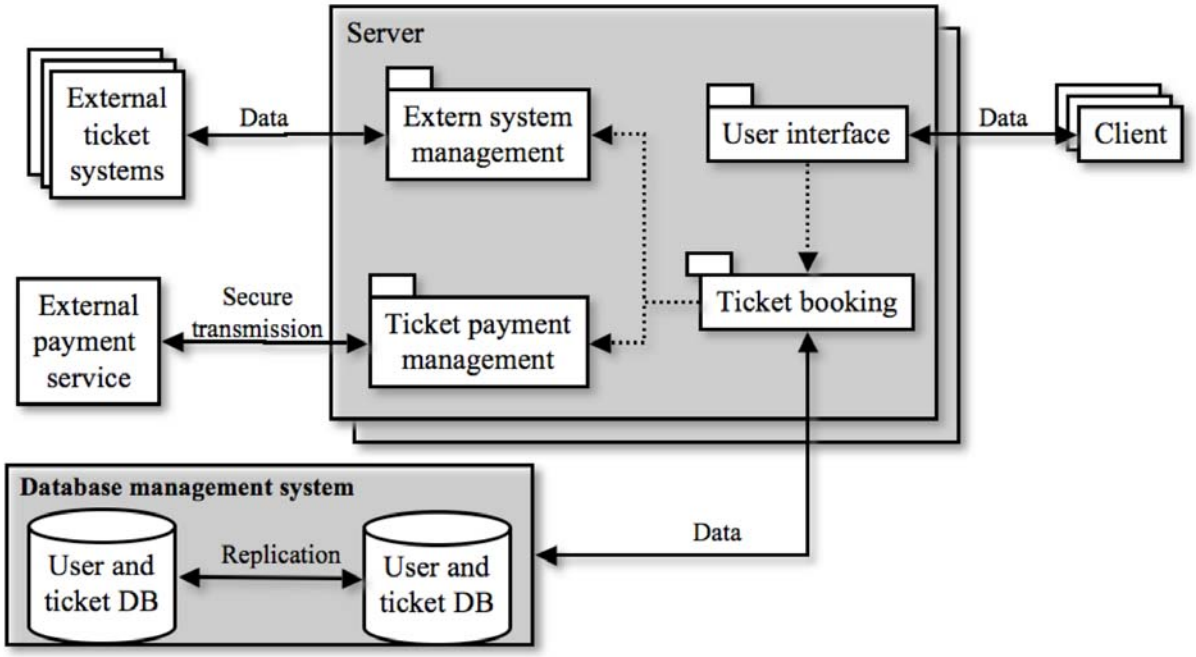
SmashYourFriends ™ is a multiplayer game where up to 5000 players (on the same server) can play against each other using their personal computer as clients. The player can choose between different vehicles like tanks, trucks, jeeps, helicopters and planes, and the goal is to shoot all the other players and to be the last survivor. All the players connected to the same server will play in the same play area. Different servers will provide different play areas like desert, arctic, city, mountain, moon, etc. An illustration of the game is shown below.

Problem 4: ATAM (10 points)

Read the description below (including the figures of the architecture and the utility tree) and *carry out an analysis of the architectural approaches for the two most important quality scenarios* (step 6 in the ATAM process).

The company MacroSoft [TM] wants to develop a booking system called GetTick. This system makes it possible for users to order/buy tickets over the web using GetTick. The users should be able to buy tickets from various sources like cinemas, theatres, sports arenas, and concert halls. The tickets are paid in the system using credit cards.

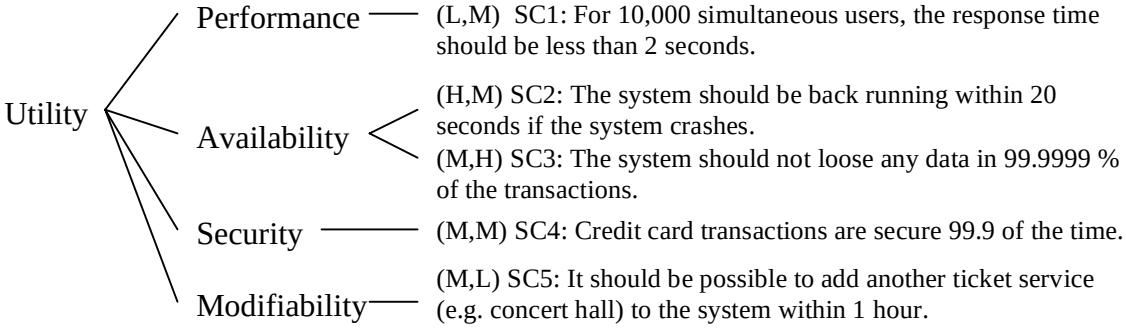
The figure below shows a logical & deployment view of the architecture of GetTick.



The following architectural tactics are a part of the architectural plan:

- AT1: Provide general interface for all external system communication.
- AT2: Use model-view controller pattern.
- AT3: Passive redundancy (warm restart).
- AT4: Replication of the database.
- AT5: Heartbeat (between primary and secondary server).
- AT6: Authenticate users.
- AT7: Data encryption.

The utility tree for the GetTick system:



Problem 5 Create an architecture (30 points)

Read the description below and do the following:

5.1 Identify the most important quality attribute(s) and the architectural drivers for the system described below (5 points)

5.2 Choose and describe suitable architectural tactics for the problem described below, and describe how the tactics affect the quality attributes (5 points)

5.3 Create architecture views of the system described below. The architecture must be described in two views according to the 4+1 view model: *Process and Logical view* (20 points)

Motivate for your choice of quality attributes, architectural drivers and the architectural tactics used in your architecture.

Software for House Alarm System BigBrother

The software described here is software for controlling an alarm system called BigBrother sold to households. The software should be able to run different configurations consisting of sensors from various producers, variations in types of displays and keyboard/button configurations.

The different configurations also represent different price segments from the very simple and cheap alarm systems to the expensive and advanced. The BigBrother software system is supports both smoke (fire) and movement sensors (theft).

In normal mode, the system is running on electrical power from a standard power socket in the wall. However, in case a power outage, the system can operate on battery power. All the sensors are powered by the BigBrother system. In case of a detection of fire or theft, the BigBrother system will start a siren (alarm sound) and the display information about what caused the alarm, in what area of the house. How the information is shown is dependent on the capabilities of the display used in the system from only simple text to graphical description of the situation.

For the more expensive configurations, the system can call the fire department or a security company through a telephone connection. The system can also be set up to call the mobile phone of the owner of the house. The system will also warn the security company if the alarm system is running on battery.

The software of BigBrother is running on custom made computer with a CPU, memory and various input/output interfaces. A physical illustration of the system is shown below.