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: 18. June

Exam in the subject
TDT4240 Software Architecture

Friday 28. May 2008
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:

Alf Inge Wang, phone 7359 4485, mobile phone: 9228 9577

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 software architecture project.

Good Luck!

Salah Uddin Ahmed and Bian Wu

Controlled 8th of May 2008

Problem 1: Various questions (20 points)

Answer these questions shortly:

1.1	What is Bass, Clements and Kazman's definition of Software Architecture (the definition in the textbook)?
1.2	What is the quality attribute usability concerned with?
1.3	Give examples of typical response measures for quality attribute performance.
1.4	How can architectural patterns and design patterns be combined?
1.5	Describe the Abstract Factory design pattern and advantages of using it.
1.6	Describe the three IT architectures presented by Sigurd Thunem (Telenor) and the areas these architectures address.
1.7	What is the most important architectural driver for commercial IT architectures according to Telenor?
1.8	What problems are addressed through the Service-Oriented Architecture (SOA) approach according to Fredrik Dahl-Jørgensen (Accenture)?
1.9	Describe the Service Identification Framework (SIF) in SOA (the main parts and their relationships).
1.10	How do different stakeholders of a software system influence the software architecture? Give examples for Developer, Marketing, Customer, End-user, and Maintainer?
1.11	Describe the difference between an architectural pattern, a reference model and a reference architecture.
1.12	Describe the three main areas of performance tactics identified by the textbook.
1.13	Why is it necessary to describe a software architecture through several views (more than one view)?
1.14	What is the purpose of evaluating the software architecture?
1.15	Describe shortly the process of reconstructing a software architecture as described in the textbook.
1.16	What is the main difference between ATAM and CBAM?
1.17	What is important to focus on when designing a software architecture for a software product line?
1.18	What are the challenges when using Off-The-Shelves (OTS) components in a software architecture?
1.19	What is the difference between a wrapper and a bridge?
1.20	What is architectural drift?

Problem 2: Choose the most appropriate architectural pattern (5 points)

Nominees:

a) Model-view-controller
b) 3-tier
c) Peer-to-peer
d) Pipe-and-filter
e) Layered
f) Blackboard

Choose the *most appropriate architectural pattern* for these 5 short descriptions of systems. Motivate for your choices:

1. A game engine that provides a high-level API to the programmer. The programmer can also access medium-level and low-level APIs to get a richer set of functionality if required.
2. An application for cooking recipes that can run on various mobile applications with various screen configuration (in size) and input devices (keys, touch-based screens, joysticks, etc).
3. A PC application for analysing weather data through a set of data transformations.
4. An application for exchanging electronic business cards between mobile devices such as mobile phones, personal data assistants (PDAs), smart phones etc.
5. A distributed collaborative application for sharing various information in a stored in common database (repository). The various parts of the application that store and retrieve information from the database should be possible to be replaced dynamically (run-time).

Problem 3: ATAM (5 points)

Do the step 6 (Analyze the architectural approaches) in the ATAM process on software for controlling a garage opening system consisting of sensors, motors with activators, and garage controller to operate the garage door.

Utility tree:

- Availability:
  - Scenario A1. The system must be available 99.9% of the time (M,H).
- Performance:
  - Scenario P1. If obstacle is detected during lowering the garage door, it must be reopened within 0.1 second (H,M).
- Security:
  - Scenario S1: It should be less than 0.01% chance to get unauthorized access to the garage controller. (L,M).

Identified architectural tactics:

- AT1: Increase computational efficiency in critical components.
- AT2: Schedule time-critical components wisely.
- AT3: Structure the system to have semantic coherence.
- AT4: Use information hiding.

Problem 4: CBAM (10 points)

From the Table 1, 2 and 3 and the information below, find:

- **Total benefit** obtained from the 3 architectural strategies.
- **Return-On-Investment** for the 3 architectural strategies
- **Rank the 3 architectural strategies** according to best investment.

Use straight lines between the data points in the graph.

The data is results from applying the Cost Benefit Analysis Method (CBAM) on a software system for managing car rentals.

Table 1: Results from prioritizing scenarios with worst, current, desired and best response levels.

Scenario	Vote	Worst	Current	Desired	Best
1. Performance: Highest number of simultaneous user requests	30	100 users	2000 users	10000 users	100000 users
2. Availability: How much of the time the server can crash	20	10% crash	1% crash	0.5% crash	0 % fail
3. Availability: Time to recover from a crash	20	10 min	3 min	1 min	0 min
4. Modifiability: Time to add support for a new type of vehicle	10	8hours	60min	10min	1min
5. Security: Probability for accessing credit card information	20	1%	0.1%	0.01%	0%

Table 2: Results from assigning utility to the various scenarios.

Scenario	Vote	Worst	Current	Desired	Best
1. Performance	30	5	40	90	100
2. Availability	20	5	50	70	100
3. Availability	20	5	30	70	100
4. Modifiability	10	5	20	50	100
5. Security	20	5	30	80	100

Table 3: Effect and cost of using architectural strategies.

Strategy	Scenario	Cost	Current response	Expected response
1. Replicated servers (hardware and software)	1 2 3	15000	2000 users 1% crash 3 min	8000 users 0.7% crash 30 sec
2. Improved computational efficiency	1	4000	2000 users	4000 users
3. Improved exception handling	2 3	3500	1% crash 3 min	0.4% crash 2 min

Problem 5 Design a software architecture using ADD (30 points)

Read the description of the Tippeliga-Ticket system below and do an architectural design using the attribute-driven design (ADD) method described in the textbook. Your answer should include:

- Architectural drivers
- Architectural tactics and patterns
- A logical view
- Interfaces
- Verification of the architecture

Note that you should only describe the logical view and only do one level of decomposition! Motivate for your choices and state your assumptions.

Tippeliga-Ticket System (TTS)

The Tippeliga-Ticket System (TTS) is a system where the users can buy tickets using credit cards to football (soccer) matches in the Tippeliga (highest division) in Norway over the Internet using a Web-browser. The user can look at information about future matches from football teams from all over Norway, and see if there are any available seats. The information about the football matches is retrieved from various servers with different interfaces provided by the different teams. Note that the teams in the Tippeliga will change every year. It is critical that the TSS is available to the users all the time, and it cannot be unavailable for more than 2 minutes a week. Before important games, such as Champion League games, it is important that the system does not break down even if over 40000 users try to get tickets at the same time.