NTNU
Det skapende universitet

Department of Computer and Information Science

Examination paper for TDT4240 Software Architecture

Academic contact during examination: Professor Alf Inge Wang
Phone: +47 9228 9577

Examination date: Friday May 31st 2013
Examination time (from-to): 09:00 – 13:00
Permitted examination support material:

- 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 an English thesaurus (English-English).

Other information:

- Simple calculator or a calculator approved by NTNU is allowed.
- The exam has 5 problems giving a total of 70 points. For each problem, each question has the same weight unless otherwise stated. The remaining 30 points are credits awarded from the software architecture project.

Language: English (Answers can be given in Norwegian or English)
Number of pages: 4

Checked by:

Date Signature

Note! Students will find examination results at Studentweb. If you have questions regarding the results, you have to contact your department.

Problem 1: Various questions (20 points)

Answer these questions *briefly*:

1.1	What is Bass, Clements and Kazman's definition of Software Architecture (the definition in the textbook)?
1.2	What is idioms when we discussion terms related to design patterns?
1.3	Describe briefly the three types of view described in the textbook: Allocation, Component-and-Connector, and Module?
1.4	What is the purpose of a utility tree, and how is it used in an ATAM evaluation?
1.5	Which architectural view describes in detail how a module is coded?
1.6	Discuss whether this statement is true or false: "Every software system has a software architecture".
1.7	Name one example of an architectural pattern that fits into the type *module*, one example that fits into the type *component-and-connector*, and one example that fits into the type *allocation*.
1.8	Give five reasons why software architecture is important according to the textbook.
1.9	Briefly explain how the four contexts *Technical*, *Project life cycle*, *Business,* and *Professional* affect the software architecture according to the textbook.
1.10	Write one fully specified quality attribute scenario for availability, using the template given in the textbook.
1.11	Describe the main types of availability tactics in the textbook, and give at least one specific example of an architectural tactic for each type.
1.12	Briefly explain the quality attribute interoperability.
1.13	Describe and explain the design pattern Abstract Factory using a class diagram.
1.14	What parameters (inputs) are typically used in models for analyzing performance?
1.15	What is an architectural pattern?
1.16	What analytic models are typically used for modeling availability?
1.17	What is a tradeoff point in an ATAM evaluation?
1.18	What is the challenge with a software product line where the scope is too narrow?
1.19	Briefly describe the steps in the Attribute-driven design method.
1.20	What techniques exist to help keep the code and the architecture consistent according to the textbook?

References:

- Bass, Clements & Kazman: "Software Architecture in Practice"
- Coplien: "Software Design Patterns: Common Questions and Answers"

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

Nominees:

a) Layered
b) Broker
c) Model-view-controller
d) Pipe-and-Filter
e) Client-Server
f) Peer-to-peer
g) Service-Oriented
h) Publish-Subscribe
i) Map-Reduce
j) Multi-tier

Choose the *most appropriate architectural pattern* (one) for the 5 descriptions below. Motivate for your choices (give reasons for choosing the pattern):

1. Wants to split a system into a number of computationally independent execution structures (groups of software and hardware) such as database, business logic, web interface and client, connected by some communication media. The structure is chosen to provide a specific server environment optimized for operational requirements and resource usage.
2. Wants to set up a set of equal distributed computational entities that are connected via a common protocol to share their services and provide high availability and scalability.
3. Wants a system that can be divided into reusable, loosely coupled components that can be flexibly combined and arranged to transform between various data formats.
4. Wants a distributed system with a structure that enables that service users do not need to know the nature or location of service providers.
5. Wants a system that quickly can analyze enormous volumes of data by sorting the data and then analyzing the grouped data.

Problem 3: Edge-dominant system (8 points)

a) What is an edge-dominant system and what characteristics do edge-dominant systems have? (2 points)
b) Draw and explain the metropolis structure of an edge-dominant system (3 points)
c) What are the implications of edge-dominant systems for the software architecture? (3 points)

Problem 4: Cloud Architecture (7 points)

a) Explain the three Cloud Service Models described in the textbook: 1) Software as a Service, 2) Platform as a Service, and 3) Infrastructure as a Service (3 points)
b) What are the economic benefits of using a cloud based architecture? (2 points)
c) What is the purpose of the hypervisor in a cloud-based architecture? (2 points)

Problem 5 Design a software architecture (30 points)

Read the description below and do an architectural design. Your answer must include:

a) Architectural drivers – 2 points
b) Architectural tactics and patterns – 3 points
c) Process view – 8 points
d) Logical view – 14 points
e) Architectural rationale – 3 points

Motivate for your choices and state your assumptions.

Software for sea buoys support for navigation at sea

There exists a collection of free-floating buoys that provide navigation and weather data to air and ship traffic at sea. The buoys collect air and water temperature, wind speed, and location data through a variety of sensors. Each buoy may have a different number of wind and temperature sensors and may be modified to support other types of sensors in the future. Each buoy is also equipped with a radio transmitter (to broadcast weather and location information as well as an SOS message) and a radio receiver (to receive requests from passing vessels. Some buoys are equipped with a red light, which may be activated by a passing vessel during sea-search operations. If a sailor is able to reach the buoy, he or she may flip a switch on the side of the buoy to initiate an SOS broadcast. Software for each buoy must:

- maintain current wind, temperature, and location information; wind speed readings are taken every 30 seconds, temperature readings every 10 seconds and location every 10 seconds; wind and temperature values are kept as a running average.
- broadcast current wind, temperature, and location information every 60 seconds.
- broadcast wind, temperature, and location information from the past 24 hours in response to requests from passing vessels; this takes priority over the periodic broadcast
- activate or deactivate the red light based upon a request from a passing vessel.
- continuously broadcast an SOS signal after a sailor engages the emergency switch; this signal takes priority over all other broadcasts and continues until reset by a passing vessel.