# The University of Melbourne
# COMP90015 Distributed Systems: Principles and Paradigms
# Semester 1, 2017 Final Examination

**School of Computing and Information Systems**
**COMP90015 Distributed Systems: Principles and Paradigms**
**Reading Time:** 15 minutes
**Writing Time:** 3 hours

**Open Book Status:** Closed Book

**This paper has 3 pages including this page**

**Identical Examination Papers:** none
**Common Content:** none

| **Authorized Materials:** |
| --- |
| No materials are authorized. |

| **Instructions to invigilators:** |
| --- |
| No papers may be taken from the exam room. |

| **Instructions to students:** |
| --- |
| All answers are to be written in the script book(s) provided.<br>Attempt all questions - partial credit is available.<br>The examination is worth 60% of the subject assessment. |

**Paper to be held by Baillieu Library:** yes


*continued on next page*

**Q.1**. (**a**) [**5 marks**] Consider a non-distributed application that allows a user to conduct a long-running simulation, with the ability to monitor and control the simulation using a graphical user (GUI) interface. The GUI interface displays a large number of simulation outputs in real-time. Now consider the same application that has been split into two processes: a server process that runs the simulation, and a GUI process that allows user monitoring and control. This is now a distributed application. Give three reasons for and two reasons against the use of a distributed application with respect to this example. Make sure your reasons are all distinctly different.

(**b**) [**5 marks**] Explain what is meant by *eventual consistency* in a distributed system. You may give an example application to aid your answer. Give your reasons for and against adopting eventual consistency in a distributed system.

**Q.2**. [**5 marks**] For each of the following architectural designs, provide a definition and give a reason for and a reason against the use of the design. Draw a diagram to illustrate the architecture.

(**a**) Client/Server

(**b**) Peer-to-Peer

(**c**) Multi-server

(**d**) Proxy server

(**e**) Thin-client

**Q.3**. [**5 marks**] In the first project, a minimum connection time interval was used, whereby connections from a given IP address were only allowed to continue if there had been no connection within the last connection time interval. What purpose does this serve? What drawbacks or weaknesses does it have? Is there a better way to achieve the same functionality? If yes, explain, if no give a reason.

**Q.4**. (**a**) [**5 marks**] Suppose that a client creates a TCP Socket to a server, sends a command, and waits for a response. However the server has failed internally and does not respond, though the connection remains open. What can be done by the client to overcome this? Critically explain your answer and include discussion concerning relevant distributed system challenges.

(**b**) [**5 marks**] Explain what is meant by a *remote procedure call* or RPC. Discuss the aspects of implementing an RPC middleware such as Sun RPC. Draw a diagram to show the architectural components of the implementation.

**Q.5**. (**a**) [**5 marks**] What are the differences between the message queue paradigm and the publish/subscribe paradigm? Give an example application where the message queue paradigm would be the more appropriate choice over publish/subscribe and say why.


*continued on next page*

(**b**) [**5 marks**] What is meant by the term *overlay network* in distributed systems? What is a benefit of using an overlay network? Give an example application that could benefit and draw a diagram.

(**a**) [**4 marks**] What is the purpose of *cipher block chaining*? Describe a technique that achieves cypher block chaining.

(**b**) [**3 marks**] List and briefly explain three worst-case assumptions when designing a secure system.

(**c**) [**3 marks**] Explain what is a *digital certificate*, including what is the basic technique used to create a digital certificate, and what is a *certificate chain*.

**Q.6**. (**a**) [**3 marks**] Explain what is meant by *process migration*. Explain two major complications with process migration.

(**b**) [**2 marks**] What is the difference between a *static* location policy and a *dynamic* location policy? Give an example static policy and an example dynamic policy.

(**c**) [**2 marks**] With respect to decentralized load balancing, what is the difference between a *sender-initiated* and a *receiver-initiated* policy? Under what load conditions is the former more efficient?

**Q.7**. A typical flat file service interface is shown below:

| | |
|---|---|
| `Read(UFID, i, n)` → `Data` | Reads up to n items from position i in the file. |
| `Write(UFID, i, Data)` | Writes the data starting at position i in the file. The file is extended if necessary. |
| `Create()` → `UFID` | Creates a new file of length 0 and returns a UFID for it. |
| `Delete(UFID)` | Removes the file from the file store. |
| `GetAttributes(UFID)` → `Attr` | Returns the file attributes for the file. |
| `SetAttributes(UFID,Attr)` | Sets the file attributes. |

(**a**) [**1 marks**] What is typically done to ensure that a `UFID` is unique across different file servers?

(**b**) [**1 marks**] Why does the interface not provide an `Open(...)` primitive?

(**c**) [**1 marks**] Which of the primitives of the interface, if any, are **not** idempotent?

**END OF EXAMINATION**