How to write a Masters thesis proposal

DAT315 The Computer Scientist in Society

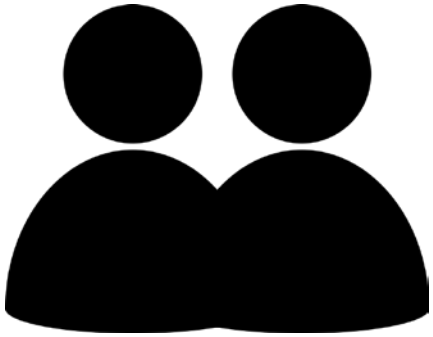
This course

- Write ”a” Masters thesis proposal
- Get feedback from course assistants
- Judged on writing

Your real Masters thesis

- No feedback
- Judged by your examiner

Same deadline!



Operating Systems
R. Stockton Gaines Editor

Time, Clocks, and the Ordering of Events in a Distributed System

Leslie Lamport
Massachusetts Computer Associates, Inc.

The concept of one event happening before another in a distributed system is examined, and is shown to define a partial ordering of the events. A distributed algorithm is given for synchronizing a system of logical clocks which can be used to totally order the events. The use of the total ordering is illustrated with a method for solving synchronization problems. The algorithm is then specialized for synchronizing physical clocks, and a bound is derived on how far out of synchrony the clocks can become.

Key Words and Phrases: distributed systems, computer networks, clock synchronization, multiprocess systems
CR Categories: 4.32, 5.29

Introduction

The concept of time is fundamental to our way of thinking. It is derived from the more basic concept of the order in which events occur. We say that something happened at 3:15 if it occurred *after* our clock read 3:15 and *before* it read 3:16. The concept of the temporal ordering of events pervades our thinking about systems. For example, in an airline reservation system we specify that a request for a reservation should be granted if it is made *before* the flight is filled. However, we will see that this concept must be carefully reexamined when considering events in a distributed system.

This work was supported by the Advanced Research Projects Agency of the Department of Defense and Rome Air Development Center. It was monitored by Rome Air Development Center under contract number F 30602-76-C-0094.

Author's address: Computer Science Laboratory, SRI International, 333 Ravenswood Ave., Menlo Park CA 94025.

A distributed system consists of a collection of distinct processes which are spatially separated, and which communicate with one another by exchanging messages. A network of interconnected computers, such as the ARPA net, is a distributed system. A single computer can also be viewed as a distributed system in which the central control unit, the memory units, and the input-output channels are separate processes. A system is distributed if the message transmission delay is not negligible compared to the time between events in a single process.

We will concern ourselves primarily with systems of spatially separated computers. However, many of our remarks will apply more generally. In particular, a multiprocessing system on a single computer involves problems similar to those of a distributed system because of the unpredictable order in which certain events can occur.

In a distributed system, it is sometimes impossible to say that one of two events occurred first. The relation "happened before" is therefore only a partial ordering of the events in the system. We have found that problems often arise because people are not fully aware of this fact and its implications.

In this paper, we discuss the partial ordering defined by the "happened before" relation, and give a distributed algorithm for extending it to a consistent total ordering of all the events. This algorithm can provide a useful mechanism for implementing a distributed system. We illustrate its use with a simple method for solving synchronization problems. Unexpected, anomalous behavior can occur if the ordering obtained by this algorithm differs from that perceived by the user. This can be avoided by introducing real, physical clocks. We describe a simple method for synchronizing these clocks, and derive an upper bound on how far out of synchrony they can drift.

The Partial Ordering

Most people would probably say that an event a happened before an event b if a happened at an earlier time than b. They might justify this definition in terms of physical theories of time. However, if a system is to meet a specification correctly, then that specification must be given in terms of events observable within the system. If the specification is in terms of physical time, then the system must contain real clocks. Even if it does contain real clocks, there is still the problem that such clocks are not perfectly accurate and do not keep precise physical time. We will therefore define the "happened before" relation without using physical clocks.

We begin by defining our system more precisely. We assume that the system is composed of a collection of processes. Each process consists of a sequence of events. Depending upon the application, the execution of a subprogram on a computer could be one event, or the execution of a single machine instruction could be one

Operating Systems
R. Stockton Gaines Editor

Time, Clocks, and the Ordering of Events in a Distributed System

Leslie Lamport
Massachusetts Computer Associates, Inc.

The concept of one event happening before another in a distributed system is examined, and is shown to define a partial ordering of the events. A distributed algorithm is given for synchronizing a system of logical clocks which can be used to totally order the events. The use of the total ordering is illustrated with a method for solving synchronization problems. The algorithm is then specialized for synchronizing physical clocks, and a bound is derived on how far out of synchrony the clocks can become.

Key Words and Phrases: distributed systems, computer networks, clock synchronization, multiprocess systems
CR Categories: 4.32, 5.29

Introduction

The concept of time is fundamental to our way of thinking. It is derived from the more basic concept of the order in which events occur. We say that something happened at 3:15 if it occurred *after* our clock read 3:15 and *before* it read

A distributed system consists of a collection of distinct processes which are spatially separated, and which communicate with one another by exchanging messages. A network of interconnected computers, such as the ARPA net, is a distributed system. A single computer can also be viewed as a distributed system in which the central control unit, the memory units, and the input-output channels are separate processes. A system is distributed if the message transmission delay is not negligible compared to the time between events in a single process.

We will concern ourselves primarily with systems of spatially separated computers. However, many of our remarks will apply more generally. In particular, a multiprocessing system on a single computer involves problems similar to those of a distributed system because of the unpredictable order in which certain events can occur.

In a distributed system, it is sometimes impossible to say that one of two events occurred first. The relation "happened before" is therefore only a partial ordering of the events in the system. We have found that problems often arise because people are not fully aware of this fact and its implications.

In this paper, we discuss the partial ordering defined by the "happened before" relation, and give a distributed algorithm for extending it to a consistent total ordering of all the events. This algorithm can provide a useful mechanism for implementing a distributed system. We illustrate its use with a simple method for solving synchronization problems. Unexpected, anomalous behavior can occur if the ordering obtained by this algorithm differs from that perceived by the user. This can be avoided by introducing real, physical clocks. We describe a simple method for synchronizing these clocks, and derive an upper bound on how far out of synchrony they can drift.

The Partial Ordering

results

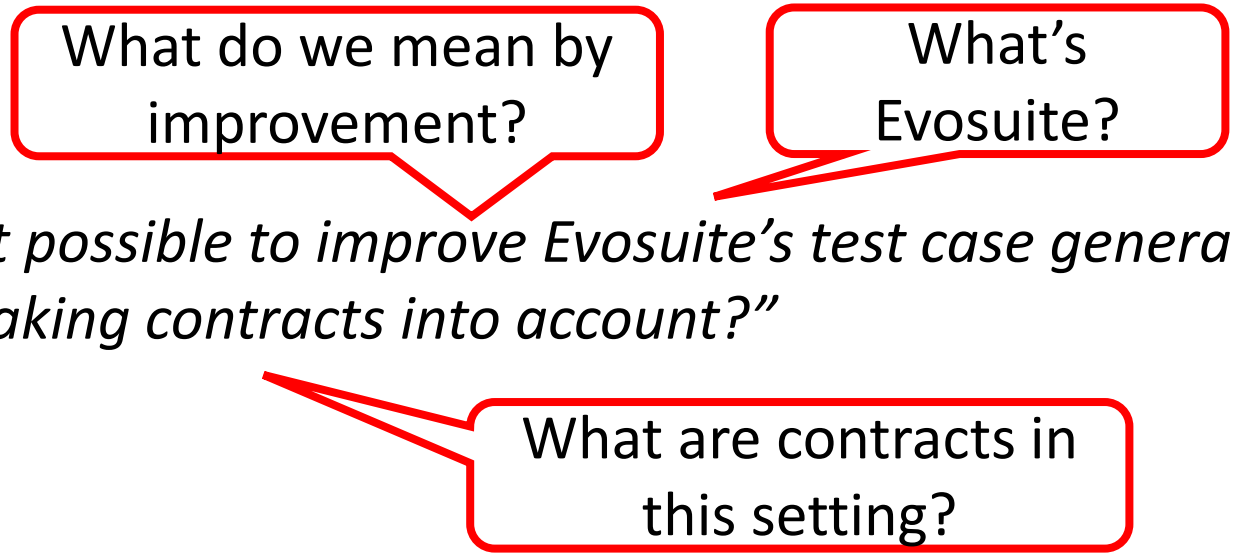
What is the problem to be solved?

- Is it interesting?
- Is it important?
- Is it feasible?

What is the problem?

- What is the question to be answered?
  - We should learn something from the project.

- There should be a *clearly defined research question* to be answered.



What is the problem?

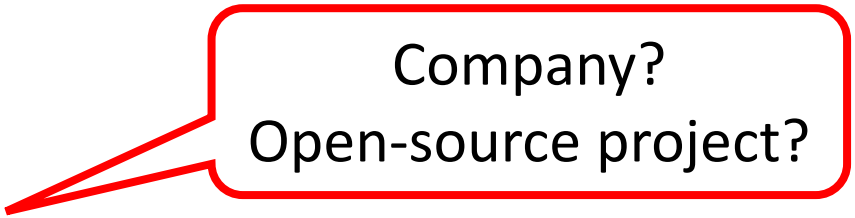
- Is it interesting?

”Is it possible to build a web site for <…> AB?”

The answer to the question should not be obvious

What is the problem?

- Is it important?
- Is there a ”customer”?
  - Why do they want it?
  - What do they plan to do with it?
- What will a solution make possible?



What is the problem?

- Is it feasible?

”I plan to prove that P=NP. Success will be the death knell of non-quantum cryptography.”

What has been done before?

- Show that you know the area
  - At the very least, identify key papers to study
- Show that there is previous work to build on
  - Feasibility!
- Identify the *gap* you will fill
  - Show what is *new*
  - Feasibility!

Problem and **previous work** are key parts of the thesis too!

What is the planned approach?

- What is your *idea*?
- What are the *key challenges?*
- Are there *initial results?*

How will you evaluate results?

How will you define success?

Simon Peyton-Jones' ***suspicious phrases:***

- "Gain insight into…"
- "Develop the theory of…"
- "Study…"

What do we mean by improvement?

”Is it possible to improve Evosuite’s test case generation by taking contracts into account?”

Why you?

- Why are *you* the right person(s) for this project?

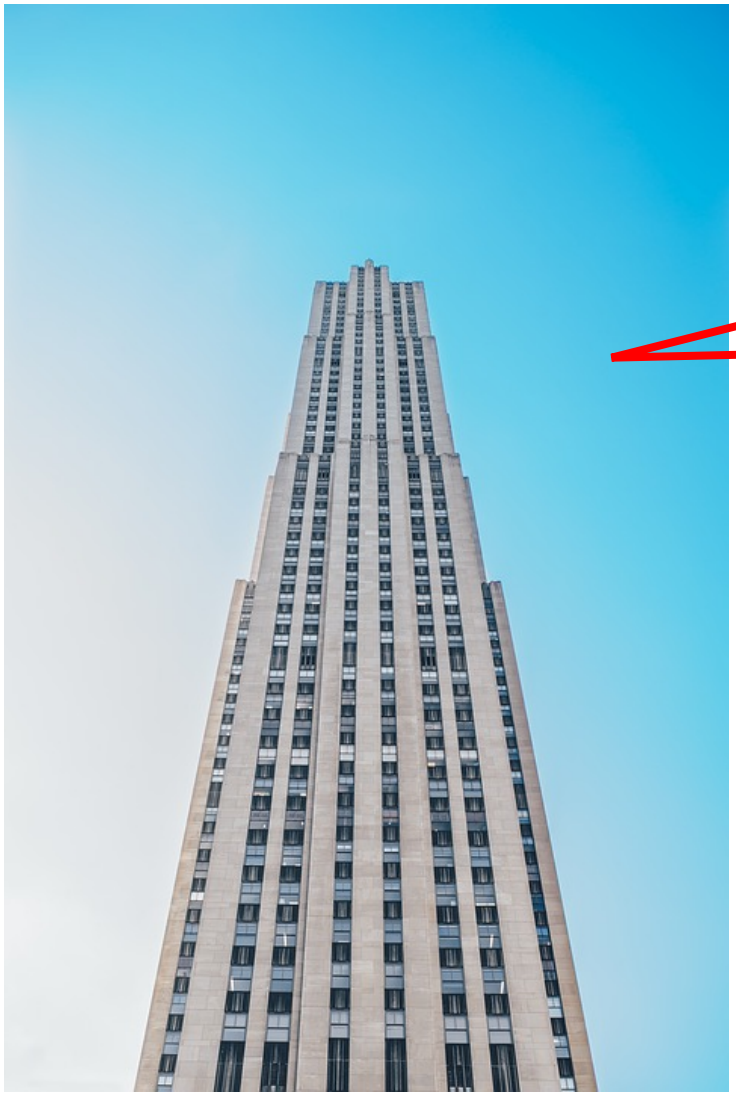
- What previous experience can you bring to bear?
  - Courses, obviously
  - Other relevant projects

Please give me money!

Please let me start my Masters thesis!

What are the risks?

- Must you learn about an entirely new tool?
  - How long will that take?
- Are you relying on existing software?
  - What if it works less well than you expect?
- What if the problem is harder than you think?



Stretch goals

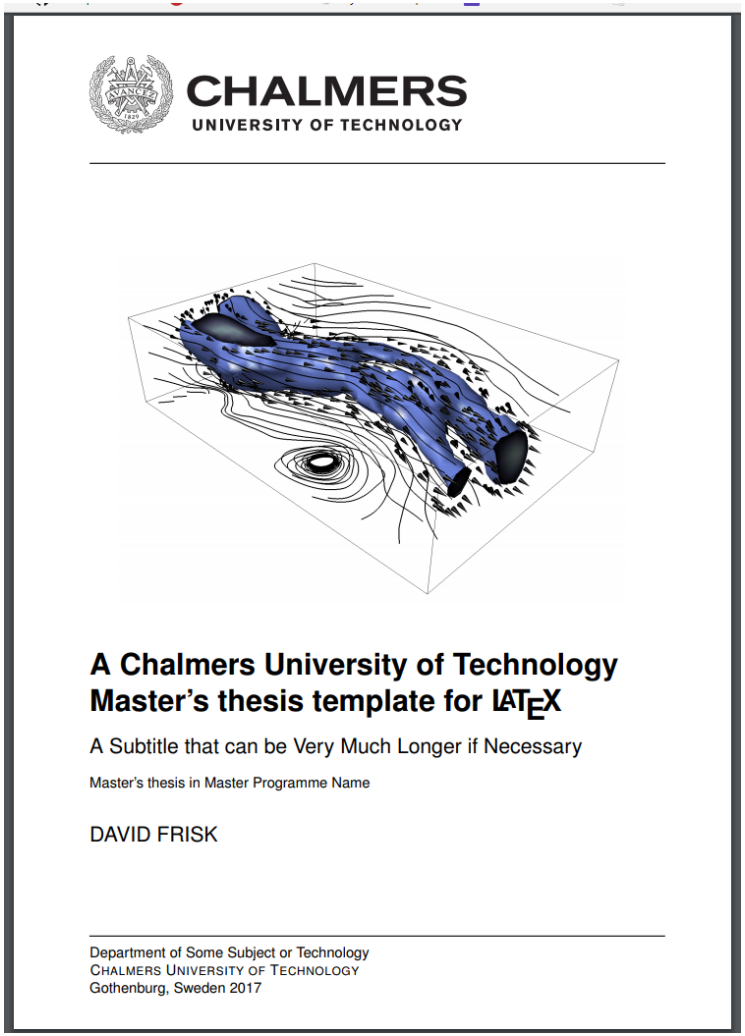
Backup plan

What about IPR?

- e.g. source code
- Open source?

- Probably hope to use the results
- May give you private information

One more thing—the template



CHALMERS
UNIVERSITY OF TECHNOLOGY

A Chalmers University of Technology Master's thesis template for LaTeX

A Subtitle that can be Very Much Longer if Necessary

Master's thesis in Master Programme Name

DAVID FRISK

Department of Some Subject or Technology
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

Contents

Theory
Methods
Results
Conclusion

Deadlines

29 Nov — Submit first draft proposal

6 Dec — You get feedback from TAs

13 Dec — Deadline for complete proposal