Lent Term 2003

http://www-compsci.swan.ac.uk/~csetzer/index.html

Dr. Anton Setzer

http://www-compsci.swan.ac.uk/~csetzer/lectures/critsys/

CS-411 Critical Systems:

http://www-compsci.swan.ac.uk/~csetzer/lectures/critsys/

(C) Anton Setzer 2003 (except for pictures)
A0. Introduction, Overview

(a) A case study of a safety-critical system failing.
(b) Two aspects of critical systems.
(c) Administrative Issues.
(d) Plan.
(e) Literature
Definition: A critical system is a computer, electronic or electromechanical system the failure of which may have serious consequences, such as

- injuries or death of human beings,
- substantial environmental damage,
- substantial financial losses,

of a critical system failing.

(a) A Case Study
Three Kinds of Critical Systems.

- **Safety-critical systems.**
  - Failure may result in the failure of the business using that system.

- **Mission-critical systems.**
  - Main topic of this module.
  - Failure may result in the failure of some goal-directed activity.

- **Business-critical systems.**
  - Failure may cause injury or death to human beings.

(C) Anton Setzer 2003 (except for pictures)
Examples of Critical Systems

- Safety-Critical
  - Critical Systems, CS-411, Lentterm 2003, Sec. A0

- Other military equipment.
  - Road traffic control (esp. traffic lights).
  - Air traffic control.
  - Railway control system.
  - Traffic control.
  - Nuclear Power Stations.
  - Chemical Industry.
  - Managed space travel
  - Military aviation.
  - Civil aviation.
  - Aerospace
  - Medical Devices.
Examples of Critical Systems (cont.)

- Navigation system of a space probe.

Mission-critical
Examples of Critical Systems (Cont.)

- Customer account system in a bank.
- Online shopping cart.
- Defense.
- Secret service.
- * Sensitive areas in companies.

* Business critical
Report from November 6, 2000

Asta Train Accident (January 5, 2000)
Figur 3.1: Rørosbanens beliggenhet i Norge. Rørosbanen markert med rød.
Kilde: Jernbaneverket

(C)AntonSetzer2003(exceptforpictures)
Figur 3.5: Rørosbanen Hamar-Støren med stasjoner, regiongrenser og grense mellom de områder togledersentralene overvåker
Kartgrunnlag: Jernbaneverket
Warning section A

Track section Aa

Track section Ab

Track section 01

Track section 02

Warning section B

Entry signal B

Tail magnet detector B

Shunting signal

Means control lamp

Point 2

Exit signal L

Exit signal N

Train route signal 1L

Train route signal 2N

Train route signal M/O

Distant signal B

Distant signal M/O

Distant signal A

Distant signal B

Distant signal L/N

Means control lamp

Means control lamp

To Elverum

To Rena

Level crossing

Figure 2-3. Rudstad station track layout
RenaTrain 230275 passengers

RudstadTrain 236910 passengers

Sequence of Events:

Railway with one track only. Therefore crossing of trains only at stations possible.

According to timetable crossing of trains at Rudstad.

Critical Systems, CS-411, Lentterm 2003, Sec. A0
Train 2302 is 21 minutes behind schedule.

Train 2302 leaves after a brief stop, Rudstad 13:06:17.

In order to cross 2369 at Rudstad.

Train 2369 leaves after a brief stop at Rena, green exit signal 13:06:15.

Leaves Rena after a stop with green exit signal, delay is reduced to 8 minutes.

3 minutes ahead of timetable.

Train 2369 leaves after a brief stop, Rudstad 13:06:17.

In order to cross 2369 at Rudstad.

Train 2302 is 21 minutes behind schedule.

10 passengers
Train 2369
Rudstad

75 passengers
Train 2302
Rena
Trains collide 13:12:35, 19 persons are killed.

The correct number hadn’t been passed on to him.

Traffic controller couldn’t warn trains, because of use of mobile telephones.

Traffic controller sees alarm approx. 13:12.

Rail traffic controller sees alarm approx. 13:07:22. Alarm signaled to the rail traffic controller (no audible signal).

Local train shouldn’t have had green signal.

10 passengers
Train 2369
Rudstad

75 passengers
Train 2302
Rena

(C) Anton Setzer 2003 (except for picture)
Figur 3.12: De to togsettene etter at brannen var slokket  
Foto: Politiet/Kripos

Figur 3.13: Motorvogn, lokomotiv, vogn nr.3 og nr. 2 etter at brannen var slokket  
Foto: Politiet/Kripos
Investigations

Traffic controller hasn’t given exit permission.

When looking again, notices that the signal has turned red.

Traffic controller surprised about the green signal.

Pulls back.

- Train has green exit signal.

- Traffic controller surprised about the green signal.

- Distinct signal green, main signal red, train drives over main signal, and

- Hanging green signal.

- Traffic controller says, he didn’t give an exit permission.

- Exit signal green and turns suddenly red.

Four incidents of wrong signals with similar signalling systems reported:

- Train driver was not blinded by sun.

- No technical faults of the signals found.

18 April 2000

Train has green exit signal.

When looking again, notices that the signal has turned red.
Several safety-critical deficiencies in the software found (some known before)

- The software used was completely replaced.

Investigations
Conclusion of SINTEF: No indication of abnormal signal status.

- Mistake of train driver (died in the accident).

Human Error.

Conclusion of SINTEF: Mistake of train driver (died in the accident).

Assessment of report by Railcert:

- CRITICISM: SINTEF was only looking for single cause faults, not for multiple causes.

(C) Anton Setzer 2003 (except for pictures)
Conclusion

It is possible that the train driver of the local train was driving against a red signal. It is possible that the train driver of the local train was not due to a software error, apparently.

Even if this particular accident was not due to a software error, apparently.

Race conditions. One possible reason for that could be a level road crossing in between and the other side of the block. The entrance signal gives the entrance signal for entrance signal into a block, for a short moment gives the entrance signal for an erroneous green exit signal due to some software error. An error that the software under certain circumstances when giving an entrance signal for the entrance signal for the entrance signal gives the entrance signal for another train and 3 minutes ahead of time, makes it likely that he received the fact that he was stopping and left almost at the same time as the other train.

Even if this particular accident was not due to a software error, apparently this software has several safety-critical errors.
In the protocol of an extremely simple installation (Brunna, 10 km from Uppsala), which was established 1957 and exists in this form in 40 installations in Sweden, a safety-critical error was found when verifying it with a theorem prover formally in 1997. Lots of other errors in the Swedish railway system were found during formal verification.

Conclusion (Cont.)
We consider a three-level model (Leveson, pp. 48-51).

**Causal Factors.**

- Level 3: Conditions and Constraints,
- Level 2: Conditions, which allowed the events on level 1 to occur,
- Level 1: Chain of events.

First Level, e.g.: that allowed the conditions on the second level to cause the events at the

- Governmental or socioeconomic policies and conditions.
- Management system, organizational culture.
- Social dynamics and human actions.
- Technical and physical conditions.

Described above.

We consider a three-level model (Leveson, pp. 48 - 51).
Root Causes

Problems found in a Level 3 analysis for the root causes.

- 1972, the cargo door latch system in a DC-10 failed, the cabin floor collapsed and only by chance the plane was not lost.
- The root cause, namely that the collapsing of the cabin floor when the cargo door opens was not fixed.
- As a consequence a fix to the cargo doors was applied.
- One DC-10 crashed 1970, killing 346 people.
- Faulty closing of the cargo door caused collapsing of the cabin floor.
- Example: DC-10 cargo-door saga.

Many examples, in which despite thorough investigation the root cause was not fixed and the accident happened again.

- If the problem behind a root cause is not fixed, almost inevitably an accident will happen again.

Root causes, weaknesses in general classes of accidents, which contributed to the current accident but might affect future accidents.

Critical Systems, CS 411, Lentterm 2003, Sec. A0-20
The driver left the station although the signal should have been red.

The local train drove over a possibly red light.

- The local train left early.

- There was no ATC (Automatic train control) installed, which stops trains.

- Software wasn't written according to the highest standards.

- Incidents had happened before, but were not investigated.

- By a software error.

- The local train probably had for short period a green light, maybe caused.

- The local train probably had for short period a green light, maybe caused.

- The local train probably had for short period a green light, maybe caused.

- The local train probably had for short period a green light, maybe caused.

- The local train probably had for short period a green light, maybe caused.

- The local train probably had for short period a green light, maybe caused.

- The local train probably had for short period a green light, maybe caused.

- The local train probably had for short period a green light, maybe caused.

- The local train probably had for short period a green light, maybe caused.

- The local train probably had for short period a green light, maybe caused.

- The local train probably had for short period a green light, maybe caused.

- The local train probably had for short period a green light, maybe caused.

- The local train probably had for short period a green light, maybe caused.
The traffic controller didn’t see the control light.

- Control panel was badly designed.
- A visual warning signal is not enough, in case the system detects a possible collision of trains.

The rail controller couldn’t warn the driver, since he didn’t know the mobile telephone number.

- To rely on a mobile telephone network in a safety critical system is extremely careless.
  * Mobile phones often fail.
  * The connections might be overcrowded.
  * Connection to mobiles might not work in certain areas of the railway network.

- The procedure for passing on the mobile phone numbers was badly managed.
The fire safety of the train engines was not very good.
Level 3 Constraints and Conditions

- Need for verified design of such software.
- Veriﬁcations to be performed for distributed algorithms.
- Flaws in the software.
- Resulting cheap solutions might be dangerous.
- Cost-cutting precedes many accidents.
- Difficult, to maintain such a small railway line.
- Controlof railway signals is a safety-critical system and should be designed with high level of integrity.
- Very diﬃcult to write correct protocols for distributed algorithms.
- Dicult to maintain such a small railway line.
- Cost-cutting precedes many accidents.
Poor human-computer interface at the control panel.

Level 3 Constraints and Conditions (Cont.)
Level 3 Constraints and Conditions (Cont.)

- The railway controller was overworked.
- Overconfidence in ICT.
- Overconfidence in ICT.
- Set up a special agreement with the mobile phone companies should have been

- Otherwise one wouldn’t have relied on the mobile phones – at least
- Otherwise one wouldn’t have used such a badly designed software.

- Flaws in management practices.

- No mechanism for dealing with incidents.
- No mechanism for dealing with incidents.

- Overconfidence in ICT.
- Overconfidence in ICT.
- Overconfidence in ICT.

- Flaws in management practices.

- No protocol for dealing with mobile phone numbers.

- Overconfidence in ICT.
- Overconfidence in ICT.
- Overconfidence in ICT.

- Flaws in management practices.

- No protocol for dealing with mobile phone numbers.

- Overconfidence in ICT.
- Overconfidence in ICT.
- Overconfidence in ICT.

- Flaws in management practices.

- No protocol for dealing with mobile phone numbers.

- Overconfidence in ICT.
- Overconfidence in ICT.
- Overconfidence in ICT.

- Flaws in management practices.

- No protocol for dealing with mobile phone numbers.

- Overconfidence in ICT.
- Overconfidence in ICT.
- Overconfidence in ICT.

- Flaws in management practices.

- No protocol for dealing with mobile phone numbers.

- Overconfidence in ICT.
- Overconfidence in ICT.
- Overconfidence in ICT.

- Flaws in management practices.

- No protocol for dealing with mobile phone numbers.

- Overconfidence in ICT.
- Overconfidence in ICT.
- Overconfidence in ICT.

- Flaws in management practices.

- No protocol for dealing with mobile phone numbers.
Lessons to be learned

• Safety-critical systems are very complex

– Cultural habits.
– Training of operators.
– Protocols the operators have to follow.
– Human-computer interaction.

Safety-critical systems are very complex:
– System aspect.

– Hardware.
– Software, which includes parallelism.

* Have to operate under adverse conditions (low temperatures, rain, light fail (light bulb of a signal might burn through), relays age.)

* Might fail (light bulb of a signal might burn through), relays age.
trains having green signals.

Both trains leave their stations on crash course, maybe caused by both trains having green signals.

* In the main example:
  - Preliminary events.
    - Events which influence the initiating event. 
  - Initiating event, triggering event.
    - Mechanism that causes the accident to occur.
  - Back to Rudstad.
  - Therefore crossing of trains moved from Rudstad to Rena.
  - Delay of the express train reduced. Therefore crossing of trains moved from express train is late. Therefore crossing of trains first moved from
  - Event.

Without them the accident cannot advance to the next step (initiating event).

= events which influence the initiating event.

A sequence of events had to happen in order for the accident to take place.
causing an accident or reduce its impact.

When designing safety critical systems, one should avoid triggering events.

Ameliorating events can prevent the accident or reduce its impact.

Intermediate events.

Events that may propagate or ameliorate the accident.

Lessons to be learned (Cont.).
Most failures of safety-critical systems were caused by multiple failures.

Preliminary and intermediate events — root cause often not fixed.

Concentration on trigger events, but not much attention to preliminary and intermediate events — root cause often not fixed.

Afterwards, concentration on trigger events, but not much attention to preliminary and intermediate events — root cause often not fixed.

For a bug was done.

The architecture of the software was investigated but no detailed search was done.

Lesson to be learned (cont.):
(b) Two Aspects of Critical Systems

Chemical industry, nuclear power industry, aviation.

- Based on techniques used in other engineering disciplines (esp.
  - Validation and verification.
  - Documentation (requirements, specifications etc.).
  - Standards.
  - etc.

- Methods for identifying hazards and measuring risk (HAZOP, FMEA)

  * Human-machine interaction.
  * Interaction with environment.
  * Hardware connected (hardware failure in sensors, relays and computers).
  * Computer system.

- System aspect.

- Software engineering aspect.

(c) Anton Setzer 2003 (except for pictures)
Failures:
Formal methods idealize a system and ignore aspects like hardware.
* Formal methods don’t replace software engineering techniques.
  – However, such methods provide verification techniques.

Need for verification techniques using formal methods.

Different levels of rigour:
(1) Application of formal methods by hand, without machine assistance.
(2) Use of formalized specification languages with some mechanized support tools.
(3) Use of fully formalized specification languages with machine assisted theorem proving.

However, such methods don’t replace software engineering techniques.
Formal methods idealize a system and ignore aspects like hardware.

Software bugs can not be avoided by careful design.

Tools for writing correct software.

Two aspects (cont.)
Two Streams in this Module

- Stream A
  - Software engineering aspect
  - General overview over formal methods.
  - Industrial practices.
Two Streams in this Module (Cont.)

Stream B (interleaved with Stream A):

○ Goal is that students have been in contact with one proof assistant.

(templates in C++ or soon in Java (??) is one approximation.)

Languages.

- Dependent types will sooner or later be used in ordinary programming

  Application of its ideas not limited to safety critical software.

  Programming with dependent types, based on Martin-Löf type theory.

  Use of the theorem prover Agda.

  Interesting: use of a theorem prover, which is used like a programming language.

  Experimental system.

  This part will be heavily machine based.

- Still area of research, a few successful industrial applications (using the

  theorem prover Coq).

- Application of its ideas not limited to safety critical software.

  Close look at one prototype example of a tool which allows to write

  application of dependent types.

- Templates in C++ or soon in Java (??) is one approximation.

- Programming with dependent types, based on Martin-Löf type theory.

- Use of the theorem prover Agda.

  Interesting: use of a theorem prover, which is used like a programming language.

  Experimental system.

  This part will be heavily machine based.

- Still area of research, a few successful industrial applications (using the

  theorem prover Coq).

- Application of its ideas not limited to safety critical software.

  Close look at one prototype example of a tool which allows to write

  application of dependent types.
Assessment:

- **80% Exam.**
  - One question concerning stream A.
  - One question mixture of stream A and B.
  - One question concerning stream B.

- **20% coursework:**
  - 4 small assignments. Each counts 5% (Plan, might be changed).
  - Mainly associated with stream B.
  - Due two weeks later.
  - Handed out approx. every 2nd week.
  - Handed in two weeks later.

Critical Systems, CS-411, Lent term 2003, Sec. A0
Course material will be continually updated.

- Web page contains overhead slides from the lectures.
- Two lectures per week.
- Monday, 13:00, Robert Recorde Room.
- Thursday, 12:00, Robert Recorde Room.

Timetable, Course Material
Learning outcome:

- Familiarity with issues surrounding safety-critical systems.
- Understanding of techniques for specifying and verifying high-integrity software.
- Experience with one proof-assistant.
- Experience with one proof-assistant.

Plan (d)
Plan Stream A

A0. Introduction, overview.
A2. Hazard and risk analyses.
A4. Fault tolerance.
A5. The development cycle of safety-critical systems.
A6. Design methods and tools.
A7. Formal methods.
A8. Verification, validation, testing.

- Probably not all topics covered (last year only A3 was reached).
Plan Stream B

B1. Introduction.
B2. The logical framework.
B3. Data types.
B4. Interactive programs in dependent type theory.
B5. Case studies.

Critical Systems, CS-411, Lenterm 2003, Sec. A0
In general, the module is self-contained.

- In the following list of books, which might be of interest, if you later have

  to study critical systems more intensively:

(e) Literature
Books Relevant for Stream A

Main course book:


Supplementary books on software-engineering aspects:


Critical Systems, CS 411, Lent term 2003, Sec. A0

A report on a lot of (100?) accidents and incidents of critical errors in software.
Some books on general formal methods:

- John Barnes: *High integrity Ada. The SPARK approach*. Has industrial applications. B-method is a specification language which might replace Z in the future.
- Practical application of the specification language Z in medical software.

Well. But probably not to be treated here.
Supposed to have a good and simple introduction into type theory.

Use of type theory in linguistics and for translation between languages.