Lecture 2 - January 8

Introduction

Lab1 Guidance
Verification vs. Validation
Mission- vs. Safety-Critical Systems

Announcements/Reminders

- Lab1 released
- Scheduled lab session tomorrow at 9am
- Office Hours: 3pm to 4pm, Mon/Tue/Wed/Thu
- Trial attendance check via iClicker today!
- Slides on Math Review (Predicates) posted
- Notes template posted
- Monday lecture venue (R N203) unchanged

Acceptance Criteria

✓ (1) Validation of Requirements → 4312.

Are we building the right product?

(a) precise → no scope of different interpretations by different engineers.

↳ . natural language (customer) poses big challenge.

. mathematics

↳ 3342: Event-b.

4315: TLA+ ↳ transition system.

(b) Complete

↳ all input scenarios should have some well-defined response by com. system

(2) Verification of Implementation

Are we building the product right?

↳ Req vs. Programs → belong to different semantic domains

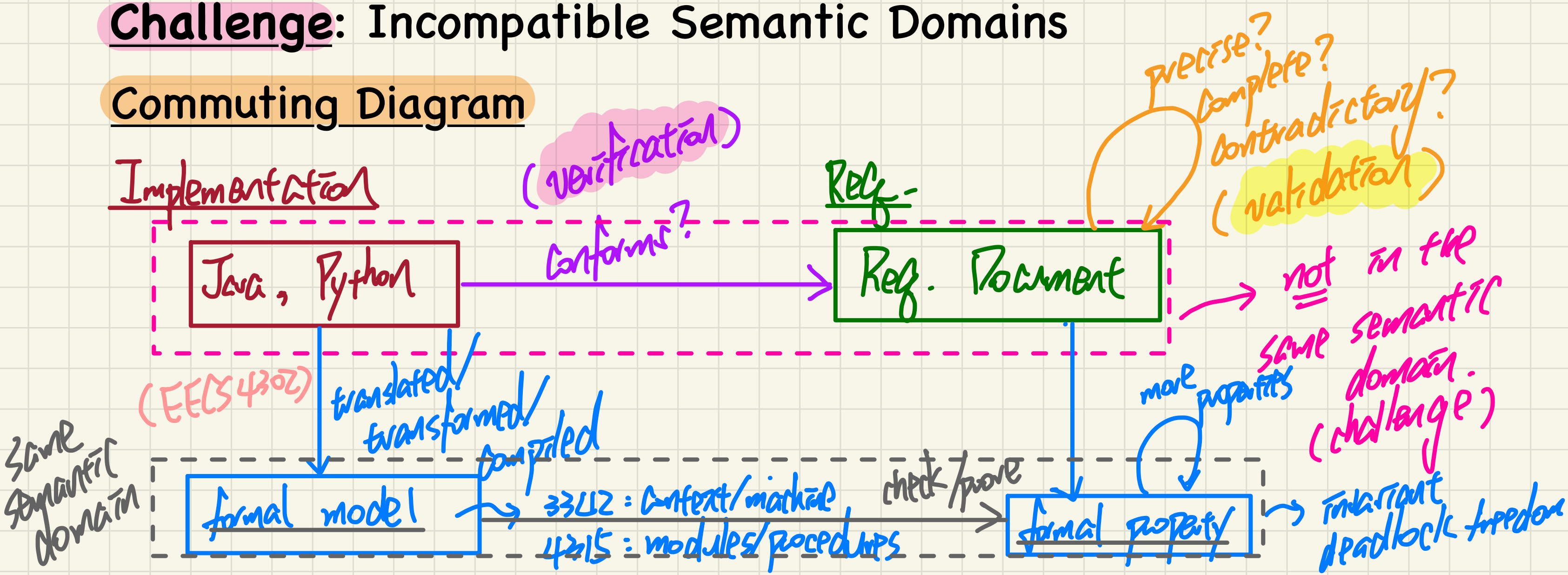
↗ solution: a implement compiler to turn them into the same semantic domain.

Goals: Verification vs. Validation

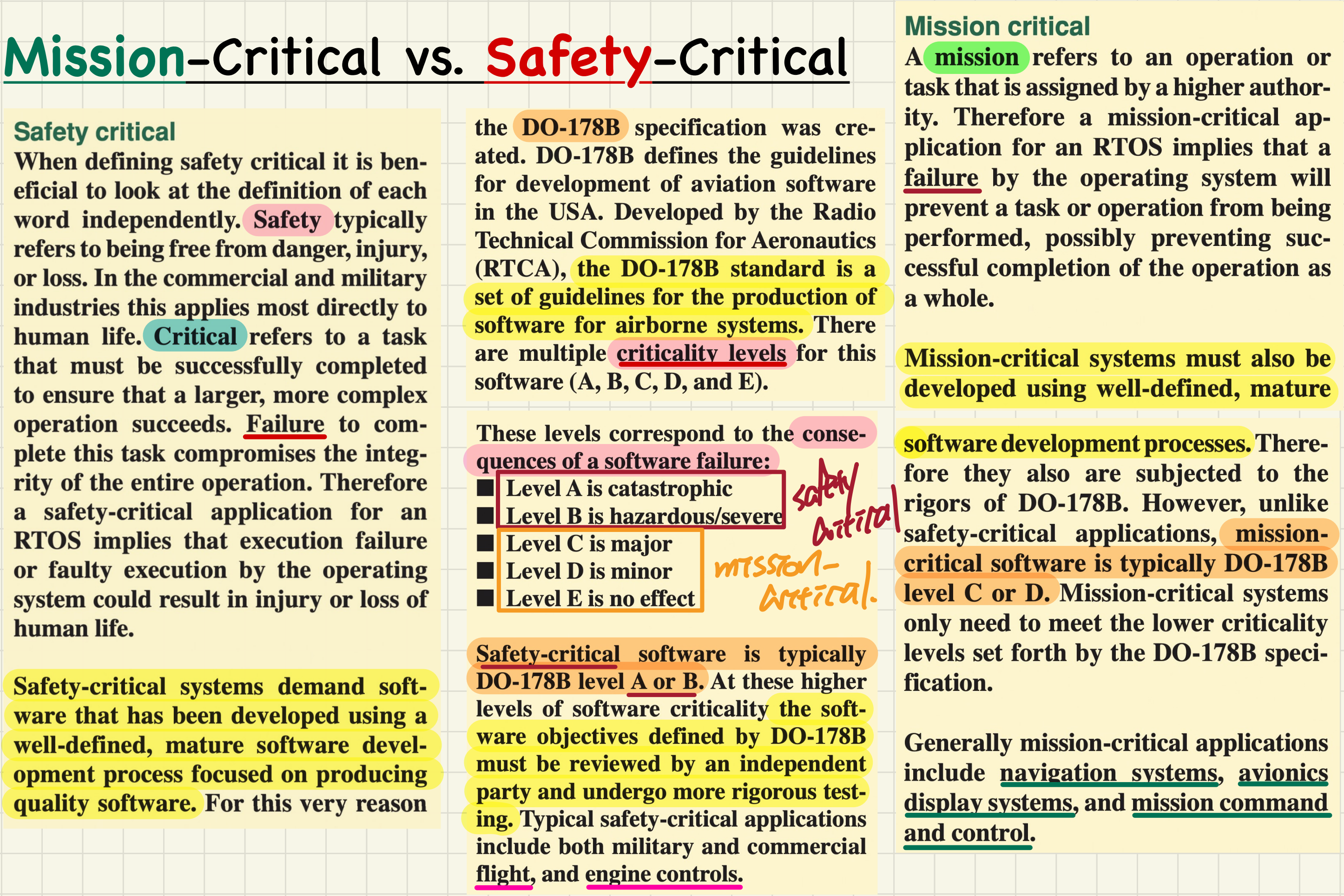
- ✓ Implementation (Java, Python)
- Requirements Document (Natural Language)
- Validity: Ambiguity, Incompleteness, Contradiction
- Compiler Technology (e.g., ANTLR4 @ EECS4302)

Challenge: Incompatible Semantic Domains

Commuting Diagram



Mission-Critical vs. Safety-Critical



Safety critical

When defining safety critical it is beneficial to look at the definition of each word independently. Safety typically refers to being free from danger, injury, or loss. In the commercial and military industries this applies most directly to human life. Critical refers to a task that must be successfully completed to ensure that a larger, more complex operation succeeds. Failure to complete this task compromises the integrity of the entire operation. Therefore a safety-critical application for an RTOS implies that execution failure or faulty execution by the operating system could result in injury or loss of human life.

Safety-critical systems demand software that has been developed using a well-defined, mature software development process focused on producing quality software. For this very reason the DO-178B specification was created. DO-178B defines the guidelines for development of aviation software in the USA. Developed by the Radio Technical Commission for Aeronautics (RTCA), the DO-178B standard is a set of guidelines for the production of software for airborne systems. There are multiple criticality levels for this software (A, B, C, D, and E).

These levels correspond to the consequences of a software failure:

- ■ Level A is catastrophic
- ■ Level B is hazardous/severe
- ■ Level C is major
- ■ Level D is minor
- ■ Level E is no effect

safety critical

mission-critical.

Safety-critical software is typically DO-178B level A or B. At these higher levels of software criticality the software objectives defined by DO-178B must be reviewed by an independent party and undergo more rigorous testing. Typical safety-critical applications include both military and commercial flight, and engine controls.

Mission critical

A mission refers to an operation or task that is assigned by a higher authority. Therefore a mission-critical application for an RTOS implies that a failure by the operating system will prevent a task or operation from being performed, possibly preventing successful completion of the operation as a whole.

Mission-critical systems must also be developed using well-defined, mature software development processes. Therefore they also are subjected to the rigors of DO-178B. However, unlike safety-critical applications, mission-critical software is typically DO-178B level C or D. Mission-critical systems only need to meet the lower criticality levels set forth by the DO-178B specification.

Generally mission-critical applications include navigation systems, avionics display systems, and mission command and control.

Source: http://pdf.cloud.opensystemsmedia.com/advancedtca-systems.com/SBS.Jan04.pdf

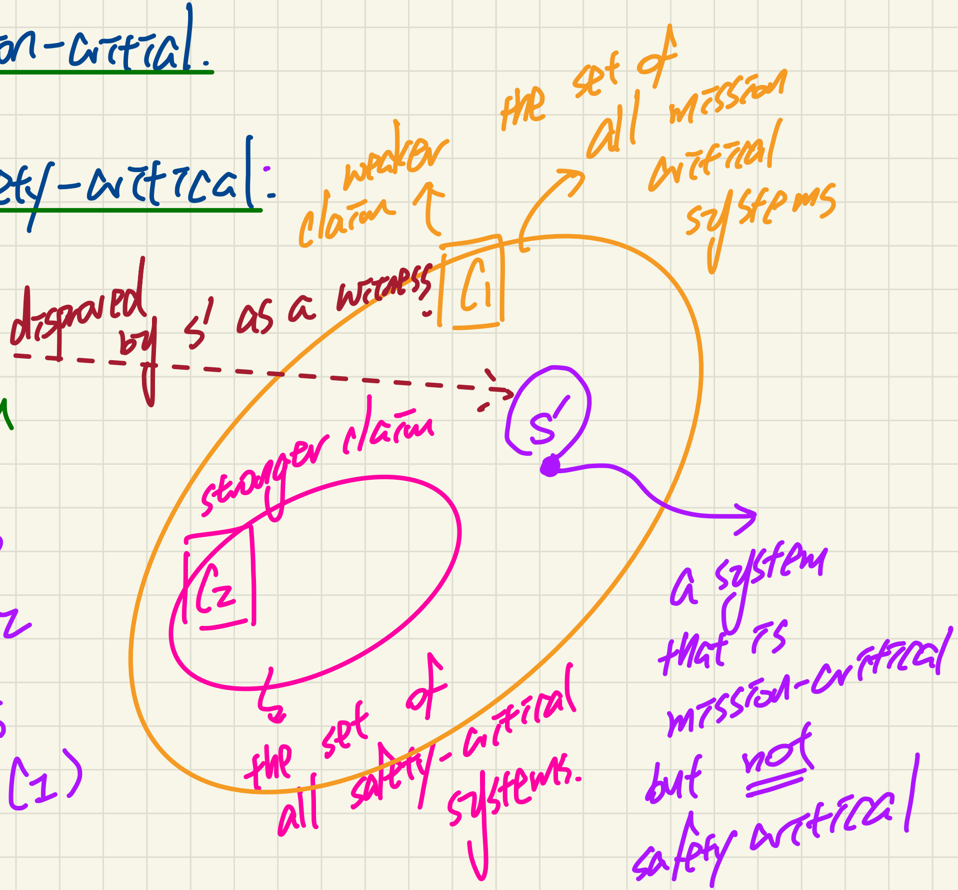
(C1) System S is mission-critical.

(C2) System S is safety-critical.

(1) $C_1 \Rightarrow C_2$ → not true in general

(2) $C_2 \Rightarrow C_1$ → C_2 is a stronger claim than C_1 (fewer systems satisfying C_2 than systems satisfying C_1)

(3) $C_1 \Leftrightarrow C_2$


disproved by S' as a witness
weaker claim
C1
the set of all mission critical systems
stronger claim
C2
the set of all safety-critical systems.
S'
a system that is mission-critical but not safety critical