IN5130 - Unassailable IT-systems

Tanusan Rajmohan - tanusanr@ifi.uio.no

UNIVERSITY OF OSLO

Autumn 2019

Contents

1 Lecture 1: Introduction to Modeling, Security and Risk **7**

1.1 This lecture aims to provide . . . 7

1.2 Part I: Classification of graphical approaches to security, risk and threat modeling . . . 7

1.2.1 What is a graphical model? . . . 7

1.2.2 What makes textual representations different from graphical? . . . 7

1.2.3 What is security? . . . 7

1.2.4 What makes graphical models for security special? . . . 8

1.3 Part II . . . 8

1.3.1 Seven Iterations . . . 8

1.4 Conclusion . . . 10

2 Lecture 2: Modeling I - Class diagrams **11**

2.0.1 Modeling a system . . . 11

3 Lecture 3: Modeling II - UML Interactions **14**

3.0.1 Causality and weak sequencing . . . 15

3.0.2 Asynchronous messages: Message Overtaking . . . 15

3.0.3 Lifeline creation and destruction . . . 16

3.0.4 Basic Sequence Diagrams Summary . . . 17

3.0.5 Summary of sequence diagrams . . . 18

4 Lecture 4: Modeling IV - UML state machines **20**

4.1 State machines . . . 20

4.1.1 Suitability of UML state machines . . . 20

4.1.2 UML State Machine . . . 20

4.1.3 An Access Control System . . . 20

4.2 Consistency . . . 22

4.2.1 Concluding the runtime consistency check . . . 23

4.2.2 Why using different states? . . . 23

4.2.3 Guidelines and Reminders . . . 23

4.2.4 What if we need to modify a state machine? . . . 24

4.3 Summarizing . . . 24

5 Lecture 5: Refinement I **25**

5.1 Three main concepts of language theory . . . 25

5.1.1 Semantic relation . . . 25

5.1.2 The need for a notion of observation . . . 25

5.1.3 Definition of a notion of observation . . . 26

5.2 Pre-post specifications, The origins of refinement . . . 26

5.2.1 Pre-post specifications . . . 26

5.2.2 Semantics of pre-post specifications . . . 26

5.2.3 Refinement in pre-post . . . 26

5.2.4 Weakening the pre-condition (assumption) . . . 27

5.2.5 Strengthening the post-condition (guarantee) . . . 27

5.3 STAIRS - Refinement in UML . . . 27

5.3.1 Motivation . . . 27

5.3.2 Traces for sequence diagrams summarized . . . 27

5.3.3 Causality and weak sequencing . . . 28

5.3.4 Weak sequencing . . . 28

5.3.5 These two diagrams are semantically the same . . . 28

5.3.6 Alternative composition . . . 28

5.3.7 Parallel composition . . . 28

5.3.8 Interaction overview diagram . . . 28

5.3.9 Dinner . . . 28

5.3.10 Some potential positive traces of Beef . . . 29

5.3.11 Potential negative Beef experiences . . . 29

5.3.12 STAIRS semantics: simple case . . . 29

5.3.13 Semantics of pre-post specifications . . . 29

5.3.14 Comparing STAIRS with pre-post . . . 29

5.3.15 STAIRS: supplementing . . . 29

5.3.16 Refinement in pre-post . . . 30

5.3.17 Supplementing in pre-post . . . 30

5.3.18 STAIRS: narrowing . . . 30

5.3.19 Indirect definition: Refinement in STAIRS . . . 30

5.3.20 Narrowing in pre-post . . . 31

6 Lecture 6: Refinement II **32**

6.1 Refinement summarized . . . 32

6.1.1 Supplementing . . . 32

6.1.2 Narrowing . . . 32

6.2 Direct definition of refinement . . . 32

6.3 Refinement formalized . . . 32

6.4 Inherent non-determinism . . . 33

6.4.1 Underspecification and inherent non-determinism . . . 33

6.4.2 The need for both *alt* and *xalt* . . . 33

6.4.3 The pragmatics of alt vs xalt . . . 34

6.4.4 Semantics - general case . . . 34

6.4.5 Notational convention . . . 35

7 Lecture 7: Refinement III **36**

7.1 Weak sequencing . . . 36

7.1.1 Weak sequencing of trace sets . . . 36

7.1.2 Weak sequencing of interaction obligations . . . 37

7.1.3 Formal semtics of seq . . . 37

7.1.4 The pragmatics of weak sequencing . . . 37

7.1.5 opt and skip . . . 38

7.1.6 The pragmatics of negation . . . 39

7.2 The pragmatics of refining interactions . . . 39

7.2.1 The use of supplementing . . . 39

7.2.2 Supplementing of interaction obligations . . . 39

7.2.3 The pragmatics of supplementing . . . 40

7.2.4 The pragmatics of narrowing . . . 41

7.2.5 The use of detailing . . . 41

7.2.6 The pragmatics of detailing . . . 42

7.2.7 The use of general refinement . . . 42

7.2.8 The pragmatics of general refinement . . . 42

8 Lecture 8: Security Risk Assessment I **43**

8.0.1 What is Security Risk Assessment? . . . 43

8.1 What is Security? . . . 43

8.1.1 Security is more than Technology . . . 43
8.1.2 Security should not be an "afterthought" . . . 43
8.2 What is Risk? . . . 43
8.2.1 Definition of Risk from ISO 31000 . . . 44
8.3 What is Risk Management? . . . 44
8.3.1 Risk Assessment involves . . . 44
8.4 Cyberspace, Cybersecurity and Cyber-risk . . . 45
8.4.1 Summary . . . 45

9 Security Risk Asessment Using CORAS **46**
9.0.1 The CORAS method . . . 46
9.0.2 The 8 Steps of the CORAS Method . . . 46
9.1 Main Concepts . . . 46
9.1.1 Definitions . . . 46
9.2 Risk Modeling . . . 47
9.3 Semantics . . . 47
9.3.1 Criticism from System Developers . . . 47
9.3.2 Criticism from Risk Analysts . . . 47

10 Lecture 9: Security Risk Assessment II **48**
10.1 The 8 steps of the CORAS method . . . 48
10.1.1 Step 1: Preparation for the assessment . . . 48
10.1.2 Step 2: Customer presentation of target . . . 49
10.1.3 Step 3: Refine target description using asset diagrams . . . 49
10.1.4 Step 4: Approval of Target Description . . . 51
10.1.5 Step 5: Risk Identification using Threat Diagrams . . . 53
10.1.6 Step 6: Risk Estimation Using Threat Diagrams . . . 53
10.1.7 Step 7: Risk Evaluation Using Risk Diagrams . . . 54
10.1.8 Step 8: Risk Treatment Using Treatment Diagrams . . . 55
10.2 Frequency calculation . . . 55

11 Lecture 10: Security Risk Assessment III **57**
11.1 Consequence calculation . . . 57
11.1.1 Pre-requisite . . . 57

11.1.2 Rule of aggregation of consequence . . . 57
11.2 Three perspective changes . . . 58
11.3 Risk Graphs with Change . . . 58
11.4 CORAS Instantiation . . . 58
11.5 Practical Example: ATM . . . 59
11.5.1 Changes . . . 59
11.5.2 Target of Analysis . . . 59
11.5.3 Focus of Analysis . . . 59
11.6 CORAS Step 5 - Risk Identification . . . 61
11.7 CORAS Step 6 - Risk Estimation . . . 61
11.8 CORAS Step 7 - Risk Evaluation . . . 62

12 Lecture 11: Uncertainty, Subjectivity, Trust and Risk How It All Fits Together 63

12.1 Uncertainty . . . 63
12.2 Subjectivity versus Objectivity . . . 63
12.3 Risk Management . . . 64
12.3.1 Definition of risk from ISO 31000 . . . 64
12.3.2 Excerise I . . . 64
12.4 Trust management . . . 64
12.4.1 Trust . . . 65
12.5 Trust versus Risk . . . 65
12.5.1 Trust aspects . . . 65
12.5.2 Risk . . . 66
12.5.3 Exercise II . . . 66
12.5.4 Trust is not inverse proportional to risk . . . 66
12.5.5 Trust is not proportional to risk . . . 66
12.5.6 Estimating risk from trust value . . . 66
12.5.7 Well-founded Trust . . . 66
12.5.8 Trustworthiness . . . 67
12.5.9 Well-founded Trust . . . 67
12.6 Three focal points in trust management . . . 67
12.6.1 Focal points illustrated . . . 67

Learning outcome

After completing the course you will be able to:

- make software that is easy to analyze with respect to security and reliability and still easy to maintain.
- understand how practical software development can benefit from theories about state machines, refinement, security risk analysis, formal reasoning, and modularity.
- perform a simple security risk analysis.

1 Lecture 1: Introduction to Modeling, Security and Risk

1.1 This lecture aims to provide

- A classification of graphical approaches to security, risk and threat modeling
- A characterization of major challenges within graphical modeling with particular focus on security, risk and threats
- Recommendations for how to deal with these challenges

1.2 Part I: Classification of graphical approaches to security, risk and threat modeling

1.2.1 What is a graphical model?

Why are you interested in graphical models for security?
(*Model -> modeling a simplified version of the system.*)

Graphical models are a marriage between probability theory and graph theory. They provide a natural tool for dealing with two problems that occur throughout applied mathematics and engineering – uncertainty and complexity ...

- From preface of Learning In Graphical Models by Michael I. Jordan (**Too Narrow**)

Wikipedia says: A graphical model is a probabilistic model for which a graph denotes the conditional dependence structure between random variables (**Too Narrow**)

1.2.2 What makes textual representations different from graphical?

Textual representations are **one-dimensional**
Graphical representations are **two-dimensional**

Definition of a graphical model
A representation in which information is indexed by two-dimensional location - J.H Larkin & H.A. Simon: 1987

1.2.3 What is security?

OR more specific: What is **cybersecurity?**
Information security: Preservation of confidentiality, integrity and availability of information (ISO/IEC 17799:2005)

Cybersecurity:

Cyber -> simplified: connected to the internet

Cybersecurity is the protection of cyber-systems against cyber-threats

A *cyber-threat* is a threat that exploits a cyberspace.

What kind of approaches for graphical modelling of security are there?

- Software engineering
  - Flow-charts → Security flow-charts (*M.Abi-Antoun et al:2007*)
  - Entity-relation diagrams → Secure UML (*T.Lodderstedt et al:2002*)
  - Use-case diagrams → Misuse-case diagrams (*G.Sindre et al:2000*)
  - State-machines → Bell–LaPadula (*W.Caelli et al:1994*)
  - Activity diagrams → UMLSec (*J.Jürjens:2004*)
  - Sequence diagrams → Deontic STAIRS (*B.Solhaug:2009*)
- Statistics/risk analysis
  - Tables → DREAD tables (*MICROSOFT:2003*)
  - Trees → Attack trees (*B.Schneier:1999*)
  - Graphs → CORAS threat diagrams (*M.S.Lund et al:2011*)

1.2.4 What makes graphical models for security special?

- Misbehaviour
- Human intensions
- Capabilities
- Defences
- Vulnerabilities
- Soft as opposed to hard constraints

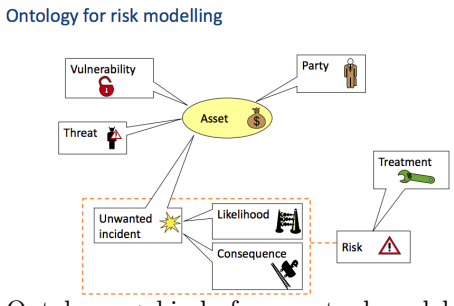
1.3 Part II

Major challenges within graphical modelling with particular focus on security, risk and threats

Recommendations for how to deal with these challenges

1.3.1 Seven Iterations

Challenge 1: Relationship to ontology



Ontology → kind of conceptual model

Make sure to avoid

- Construct deficit
- Construct overload
- Construct redundancy
- Construct excess

Challenge 2: The number of symbols?

The amount of information that is transmitted by a human being along one dimension is seven, plus or minus two (G.A. Miller:1956)

Most humans cannot reliably transmit more than

- 6 pitches (tones)
- 5 levels of loudness
- 4 tastes of salt intensities
- 10 visual positions (short exposure)
- 5 sizes of squares
- 6 levels of brightness

Fix: Use several dimensions!

Challenge 3: What kind of symbols

(D.L.Moody:2009) recommends amongst others

- Different symbols should be clearly distinguishable
- Use visual representations suggesting their meaning
- Include explicit mechanisms to deal with complexity
- Include explicit mechanisms to support integration
- Use the full range of capacities of visual variables

Be aware of the theory of gestalt psychology

- Law of proximity
- Law of similarity
- Law of closure
- Law of symmetry
- Law of common fate
- Law of continuity
- Law of good gestalt
- Law of past experience

Challenge 4: Semantics

What is a semantics? Why do we bother to define semantics?

- You need more than one semantics
- Start by defining a natural language semantics
- Make sure the semantics works for incomplete diagrams
- Be careful with hidden constraints
- The ability to capture inconsistencies is often a good thing

Challenge 5: Documenting consequence

When I was young and stupid I measured any loss, impact or consequence in monetary value (That's not a good idea!)

Fix

- Define assets carefully
- Decompose or try to avoid fluffy assets
- Define concrete scales for each asset

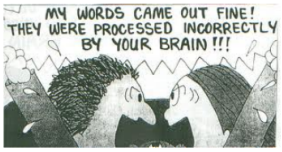
Challenge 6: Documenting likelihood

Bad communication: Probability (G. Gigerenzer:2002)

- "30-50% probability for sexual problems if you take for Prozac" means ...
  - of 10 times you have sex, you will get problems in 3-5?
  - of 10 patients, 3-5 will get problems?
  - ...

Bad communication: Probability

- Implicit reference – invites missunderstandings
- Fix: Use frequencies
  - "Of 10 patients 3-5 will get sexual problems"



Challenge 7: Documenting risk

Bad communication: Relative risk (G. Gigerenzer:2002)

- "People with a high level of colestreaol may reduce their risk of death by 22 % by taking medicine X"
- Basis for statement (Treatment in 5 years):

Treatment	# deaths pr 1000 with high colestreaol
Medicine X	32
Placebo	41

$$\frac{41-32}{41} = 22\%$$

- Often missunderstood as follows: "If 1000 persons with high colestreole takes medicine X, 220 will be saved."
- Fix: Formulate as absolute risk reduction:
  - Medicine X reduces the number of deaths from 41 to 32 per 1000.
  - The absolute risk reduction is 9 per 1000, i.e. 0,9 %.

1.4 Conclusion

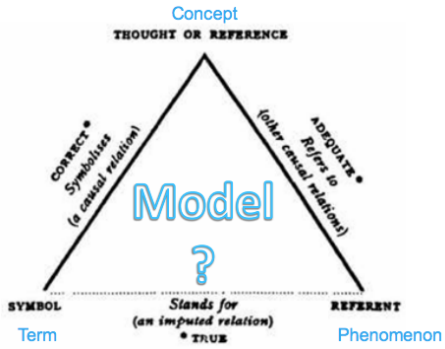
The form of representations has an equal, if not greater, influence on cognitive effectiveness as their content (D.L. Moody:2009)

Refinements -> relationship between the model and the system

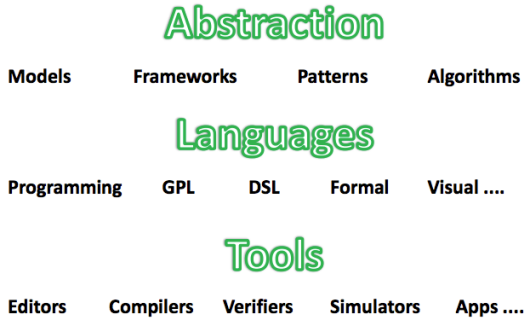
Security risk assessment -> model the backside of the model (Coras).

2 Lecture 2: Modeling I - Class diagrams

What's a model?



Artefacts in Informatics



2.0.1 Modeling a system

A system is a part of the world

- which we *choose* to regard as a whole, *separated* from the rest of the world *during some period* of consideration, a whole which we choose to consider as containing a *collection of components*, each characterized by a selected set of *associated data* items and patterns, and by *actions* which may involve itself and other components

Mental systems

- System existing in the human mind, physically materialized as states of the cells of our brains

Mental and manifest models

- When a limited set of properties is selected from a system

These definitions are from K. Nygaard and his DELTA team (in 1977)

What language(s) to use?

The language must have good mechanisms for abstraction, must have adequate tooling and must scale to "real systems"

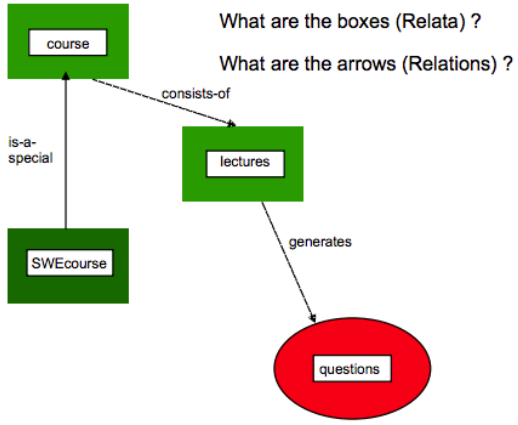
UML Class Modeling: Concepts, Identity, Generation, Meta, Aggregate

Concepts: Class, Type, Pattern, Method, Function, Datatype, Object, Instance, Entity, Method call, Function call, Variable, Prototype, Clone

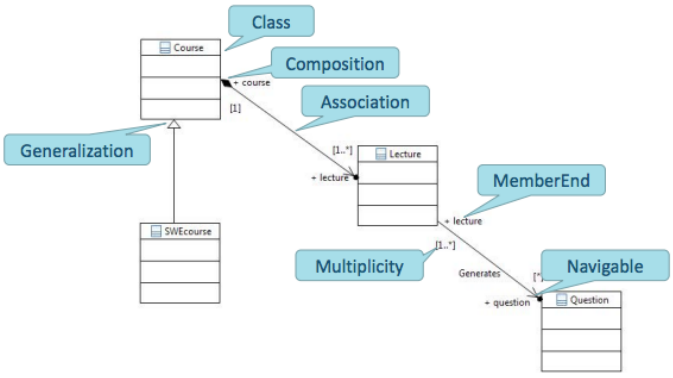
Example: A small story about Courses

The Software Engineering Course is a special Course. Courses contain Lectures. The lectures may generate questions

A small Story with Boxes and Arrows

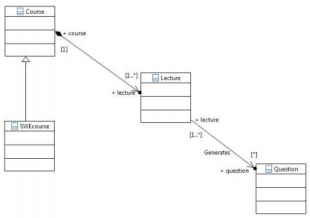


A small Story with UML class diagram



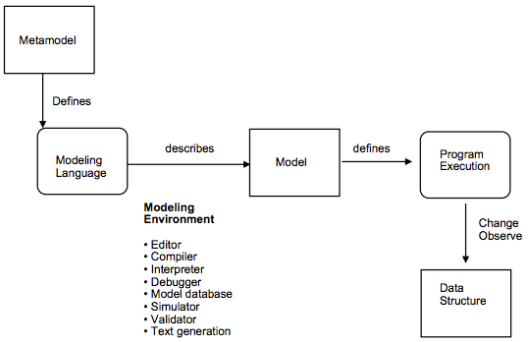
Exercise:

- Can Software Engineering course (SWEcourse) be held without lectures?
- Can there be lectures without questions asked?
- Can the very same lecture be given in two different courses?
- Can the very same question be posed to several lectures?
- If a course is cancelled, will all remaining lectures also be cancelled? (or "terminated")



Identity modifiers:	**Languages:**
Generalization Subclass Derived Classes Extension	UML Simula C++ Java
Interface	UML, Java
Parameters Overloading	FORTRAN, Pascal, Algol, ... C++, Java
Redefined operations Virtual procedures Virtual functions Overriding methods	UML Simula, Smalltalk C++ Java
Pointers to functions	C, C++

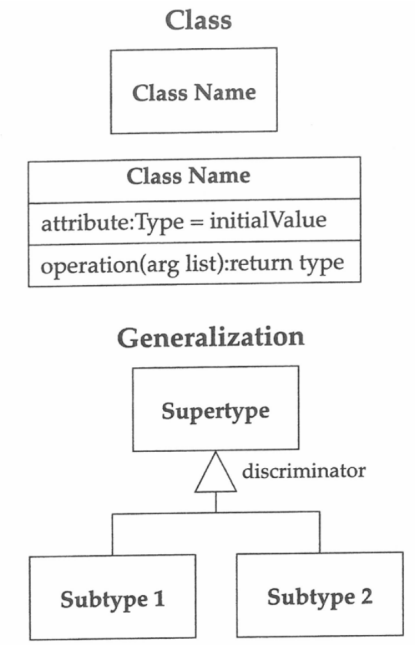
Subclassing or Inheritance

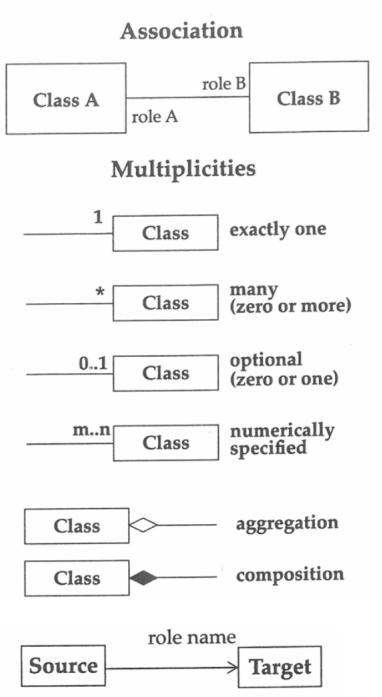


Generation

Lev	UML model	Language	Programming	Language
M3	MOF metamodel	MOF	Grammar of BNF	BNF?
M2	UML metamodel	MOF	Grammar of Java	BNF
M1	UML user model	UML	Java user program	Java
M0	Execution of user model		Execution of java program	

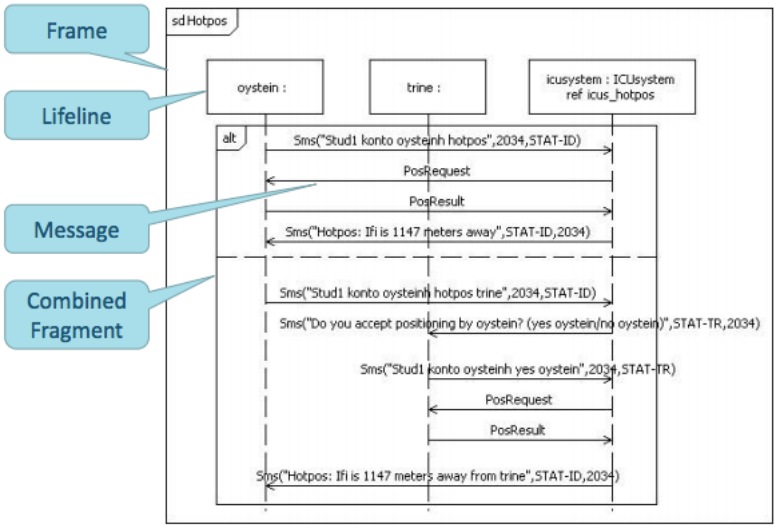
Class Diagram Summary





3 Lecture 3: Modeling II - UML Interactions

(also called Sequence Diagrams) - This is a Sequence Diagram



Sequence diagram in a nutshell

They are simple, powerful, readable and they emphasize the interaction between objects when interplay is the most important aspect. Often only a small portion of the total variety of behavior is described to improve the individual understanding of an interaction problem.

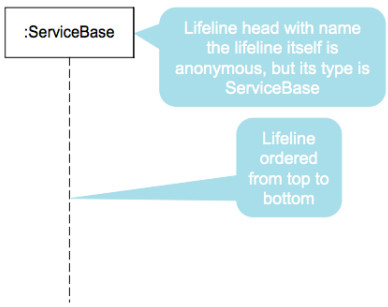
Sequence Diagrams are used to: document protocol situations, exemplify behavior situations, verify interaction properties relative to a specification, describe test cases, document simulation traces.

The example context: Dolly Goes To Town

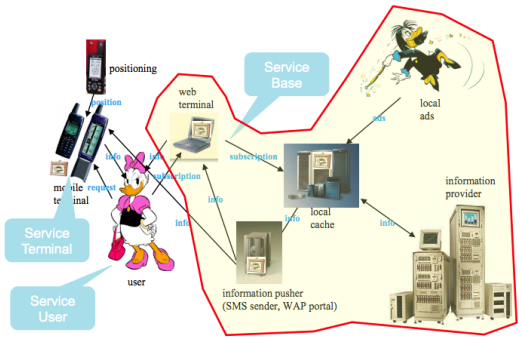
Dolly is going to town and wants to subscribe for bus schedules back home given her current position and the time of the day.

The service should not come in effect until a given time in the evening

Lifeline – the "doers"

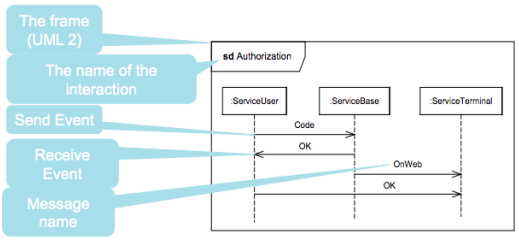


The informal architecture



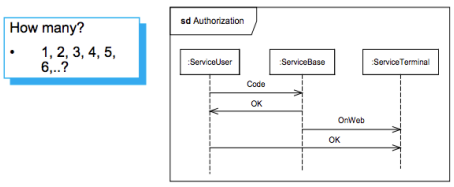
(Simple) Sequence Diagram

- Messages have one send event, and one receive event.
  - The send event must occur before the receive event.
- Events are strictly ordered along a lifeline from top to bottom



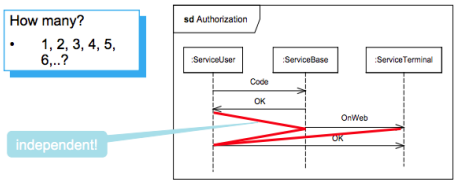
Exercise: How many *global traces* are there in this diagram?

- The only invariants:
  - Messages have one send event, and one receive event. The send event must occur before the receive event.
  - Events are strictly ordered along lifeline



How many global traces are there in this diagram?

- The only invariants:
  - Messages have one send event, and one receive event. The send event must occur before the receive event.
  - Events are strictly ordered along lifeline



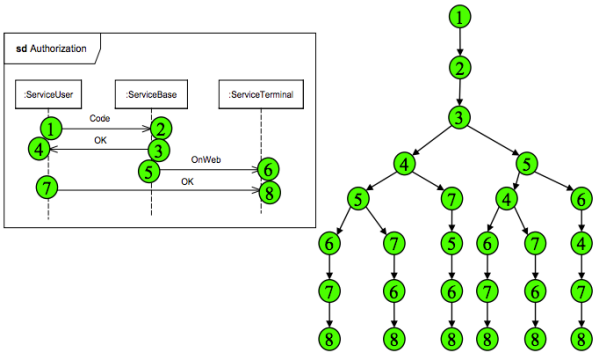
3.0.1 Causality and weak sequencing

Causality:

- a message can never be received before it has been transmitted
- the transmission event for a message is therefore always ordered before the reception event for the same message

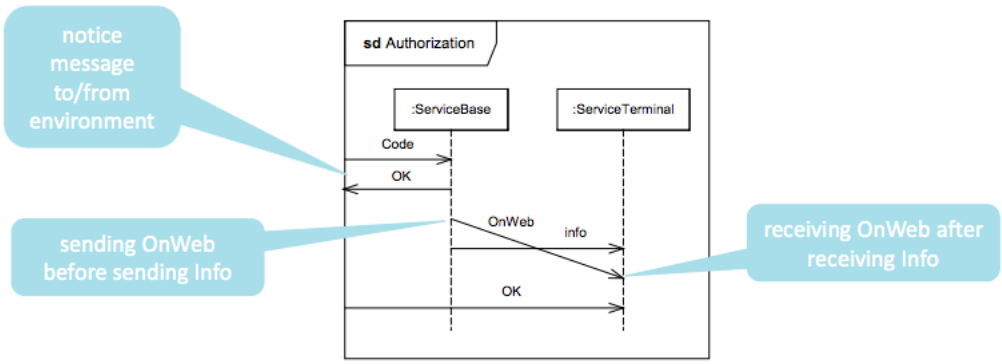
Weak sequencing:

- events from the same lifeline are ordered in the trace in the same order as on the lifeline

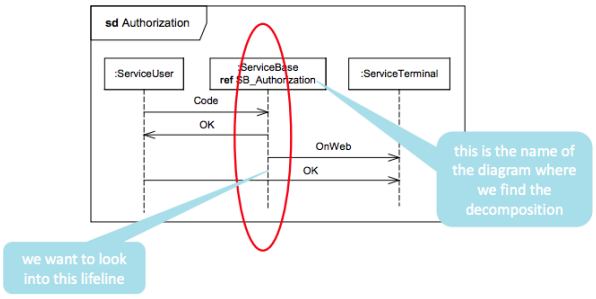


3.0.2 Asynchronous messages: Message Overtaking

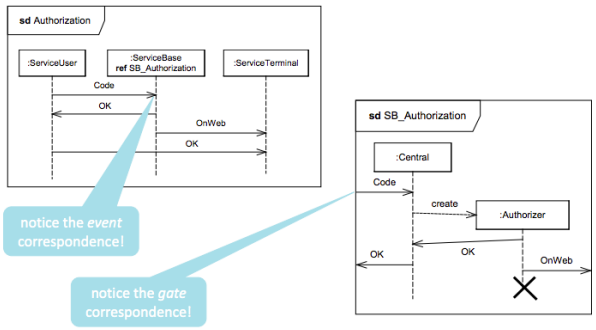
- asynchronous communication = when the sender does not wait for the reply of the message sent
- Reception is normally interpreted as consumption of the message.
- When messages are asynchronous, it is important to be able to describe message overtaking.



Decomposing a Lifeline relative to an Interaction

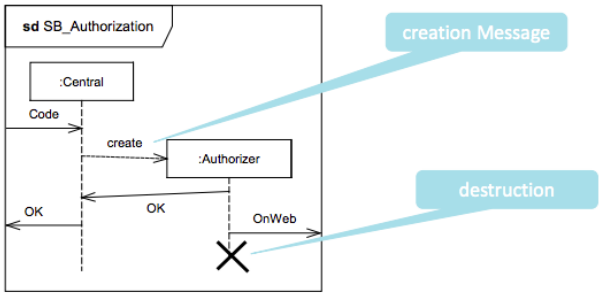


The Decomposition

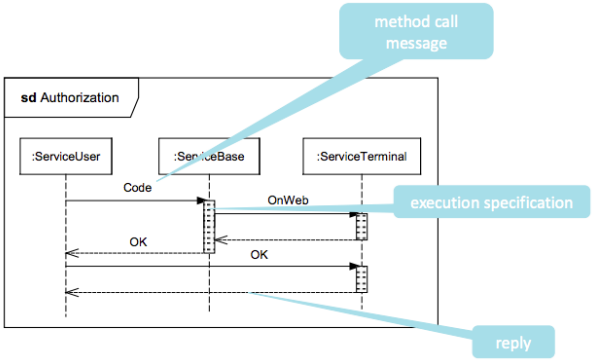


3.0.3 Lifeline creation and destruction

- We would like to describe Lifeline creation and destruction
- The idea here (though rather far fetched) is that the ServiceBase needs to create a new process in the big mainframe computer to perform the task of authorizing the received Code. We see a situation where several authorizers work in parallel



Synchronizing interaction



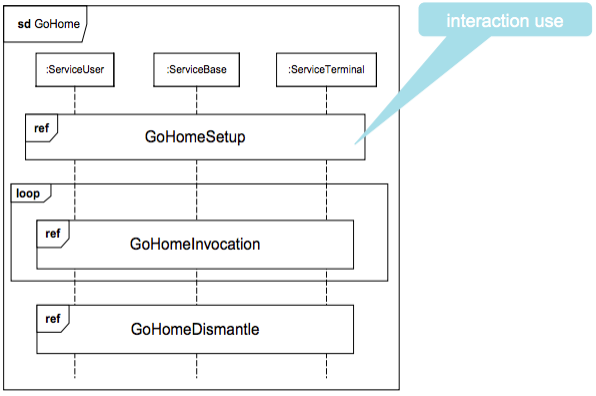
3.0.4 Basic Sequence Diagrams Summary

- We consider mostly messages that are *asynchronous*, the sending of one message must come before the corresponding reception
- UML has traditionally described *synchronizing* method calls rather than asynchronous communication
- The events on a lifeline are strictly *ordered*
- The *distance* between events is not significant
- The *context* of Interactions are classifiers
- A lifeline (within an interaction) may be detailed in a *decomposition*
- Dynamic *creation* and *destruction* of lifelines

More structure

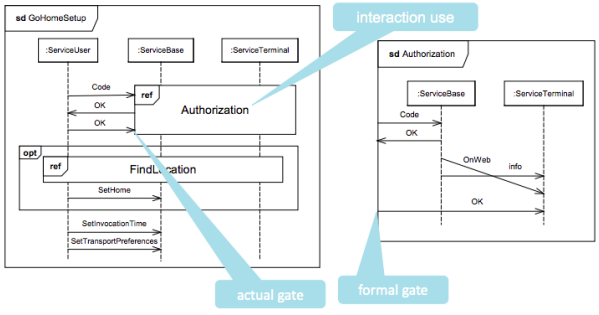
- **interaction uses** - such that interactions may be referenced within other interactions
- **combined fragments** – combining Interaction fragments to express alternatives, parallel merge and loops
- **better overview of combinations** – High level Interactions where Lifelines and individual Messages are hidden
  - Not so useful since no tools support this
- **gates** – flexible connection points between references/expressions and their surroundings
  - we have looked at this in the context of decomposition, but gates are also on InteractionUse and CombinedFragments

References

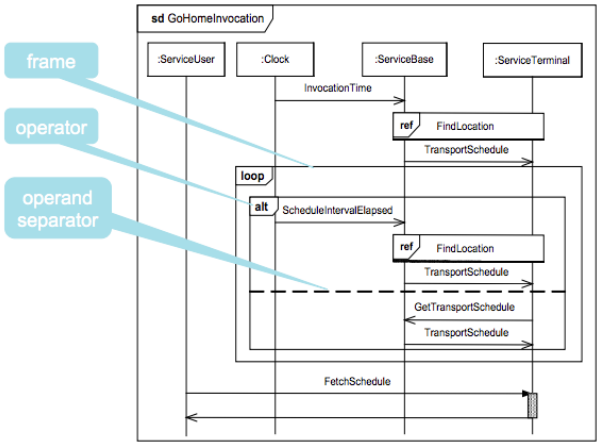


Gates

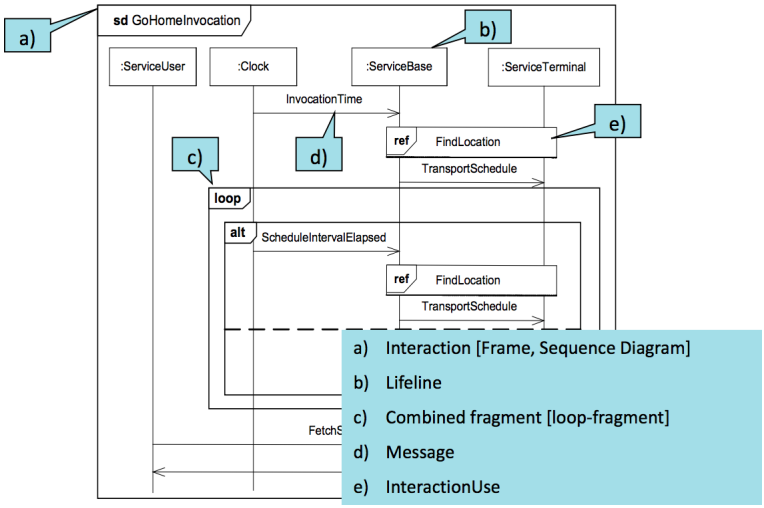
The formal gates are the ones inside authorization and the actual gate is the one pointing in to the authorization diagram



Combined fragment example

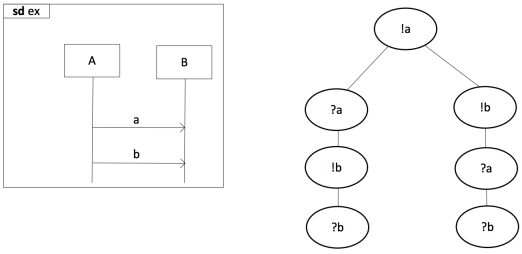


And now chiefly yourselves !!!

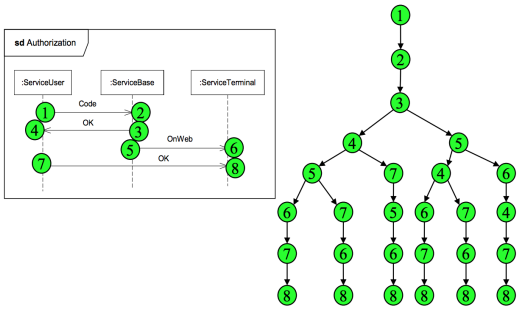


3.0.5 Summary of sequence diagrams

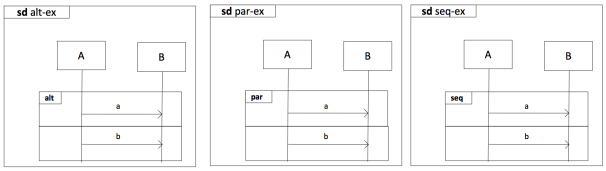
positive behavior I - 4 traces



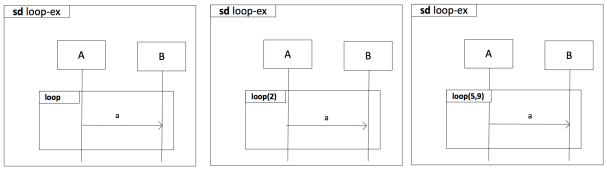
positive behavior II - 8 traces



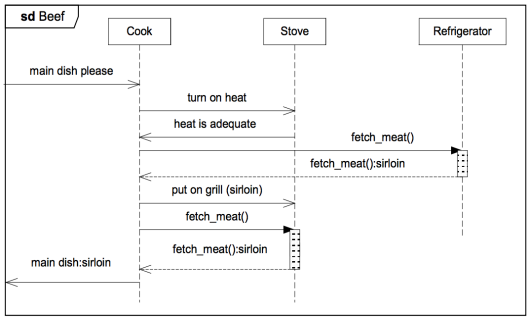
positive behavior III - 4 traces



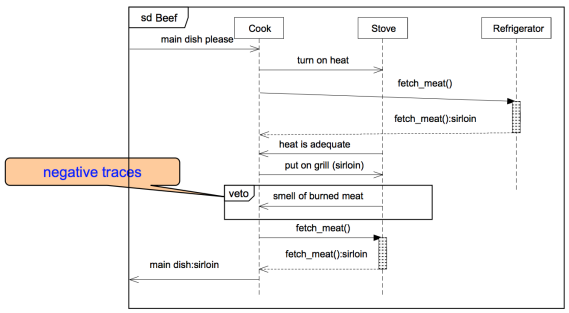
positive behavior IV



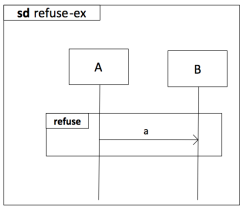
Ordering Beef



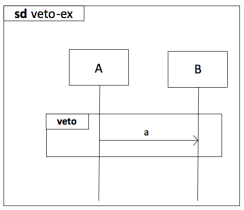
Ordering Beef also including negative behavior



veto and refuse

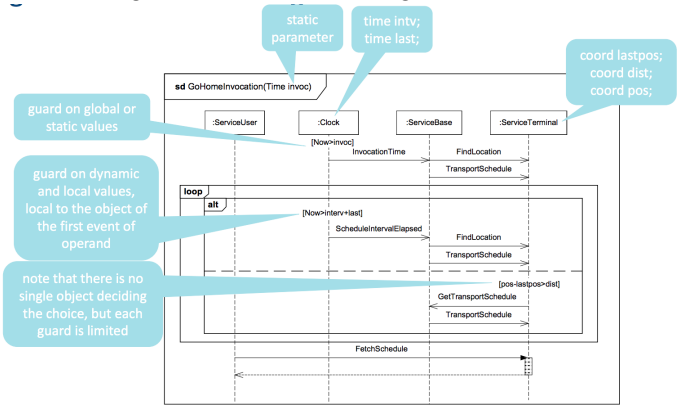


({},{<!a,?a>})



({<>},{<!a,?a>})

Negative behavior due to guards

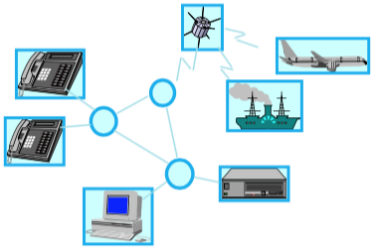


4 Lecture 4: Modeling IV - UML state machines

4.1 State machines

4.1.1 Suitability of UML state machines

- reactive
- concurrent
- real-time
- distributed
- heterogeneous
- complex



4.1.2 UML State Machine

Finite

- a finite number of states

State

- a stable situation where the process awaits stimuli
- a state in a state machine represents the history of the execution

Machine

- that only a stimulus (signal, message) triggers behavior
- the behavior consists of executing transitions
- may also have local data

Exercise
What is a state in a programming language?
- A state is the current value of my programming variables
What is a machine in a programming language?
- A machine is a method in programming language

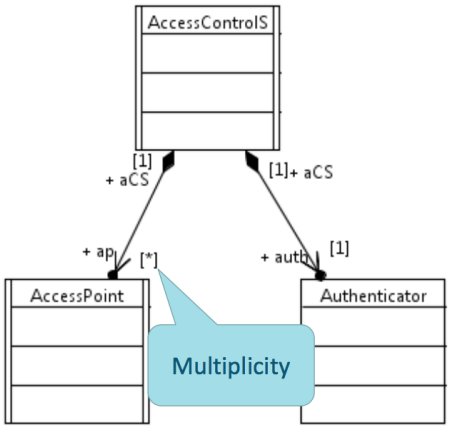
4.1.3 An Access Control System

- A set of Access Points are established to control the access to an area
- The Access Points controls the locking of a door
  - in a more abstract sense, access control systems may control bank accounts or any other asset that one wants to protect
- The Access Point access is granted when two pieces of correct identification is presented

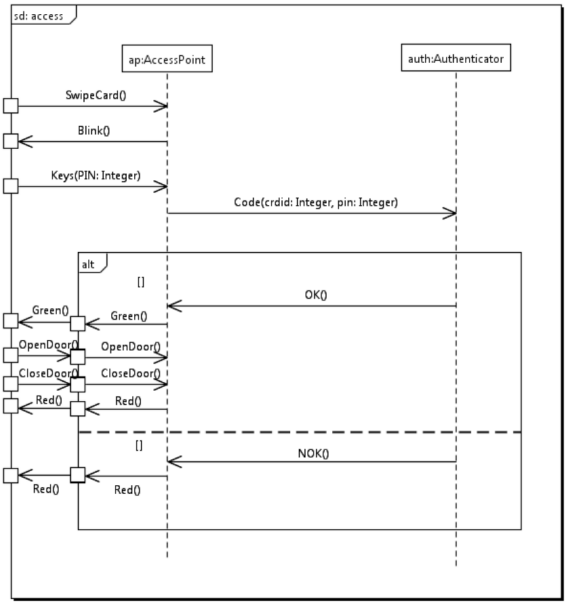
  - A card
  - A PIN (Personal Identification Number)

- The access rights are awarded by a central Authentication service

The concepts in a class diagram



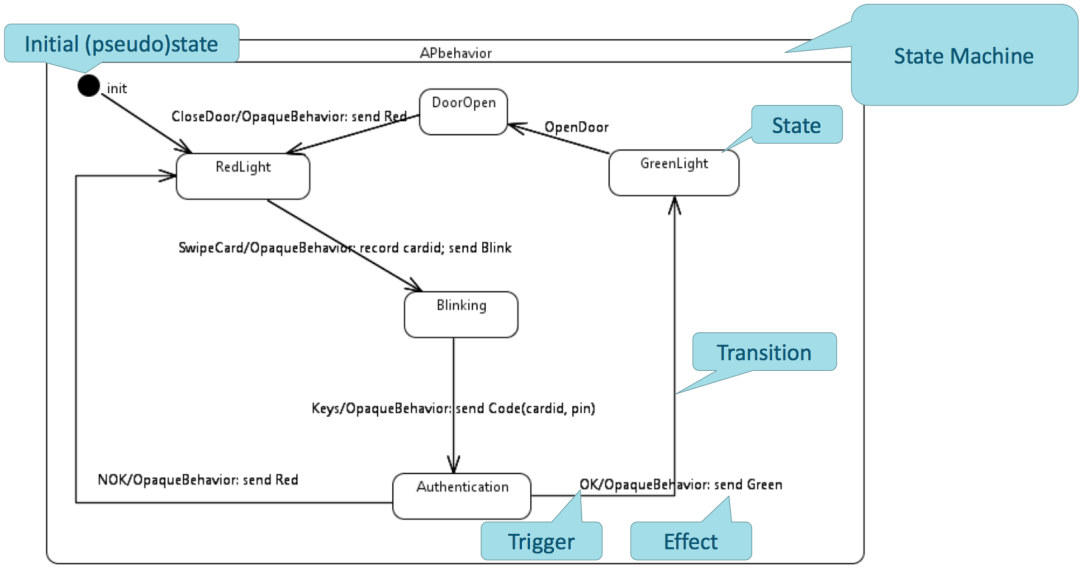
Happy Day Scenario



OpaqueBehavior is a UML behavior defined in another language

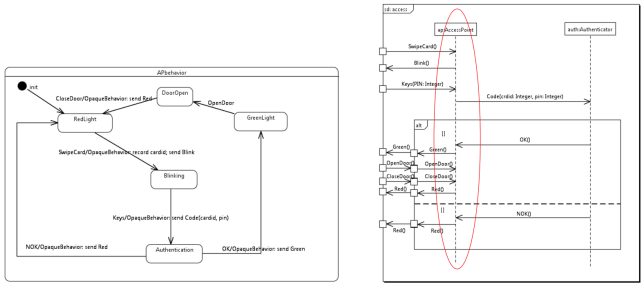
In this course we are flexible wrt how behaviors are expressed Hence, using the OpaqueBehavior construct is not important

The behavior of the AccessPoint

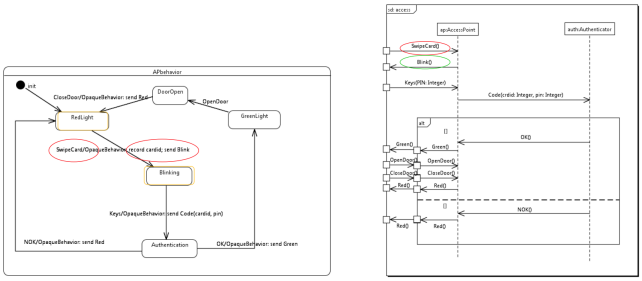


4.2 Consistency

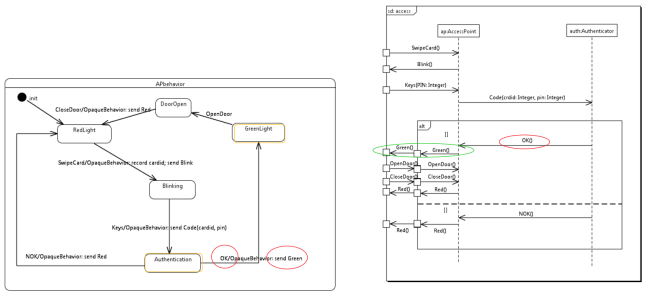
Runtime consistency – behaviors corresponding



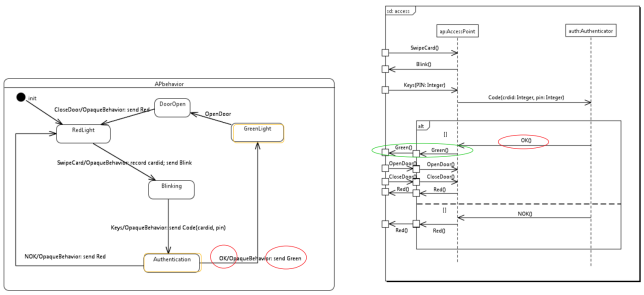
Let's execute the state machine according to the sequence diagram



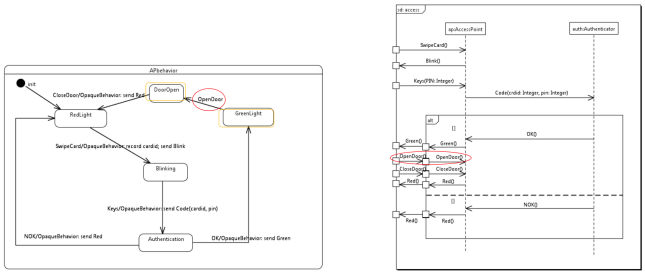
Play it again Sam



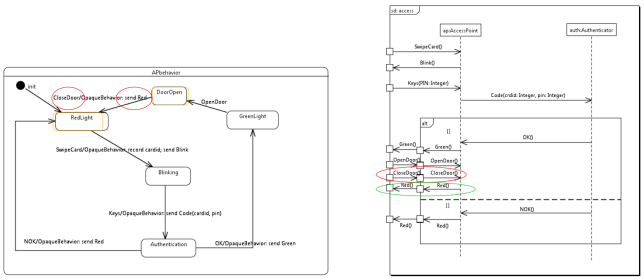
Access granted (one out of two alternatives)



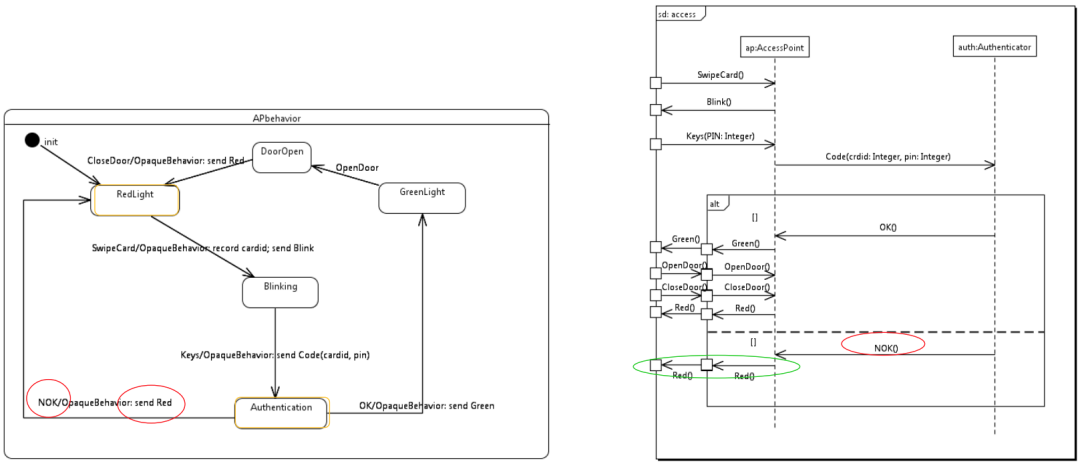
User opens the door



User closes the door again



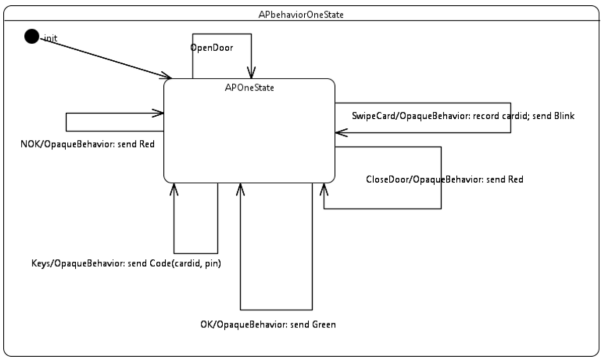
Access not granted (second of two alternatives)



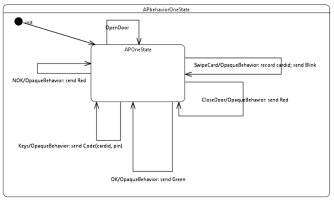
4.2.1 Concluding the runtime consistency check

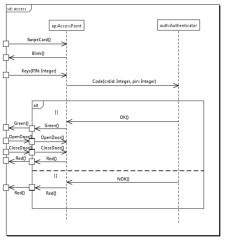
- The APbehavior state machine satisfies all traces of the sequence diagram access
- Thus these behaviors are consistent

Another attempt to define the state machine

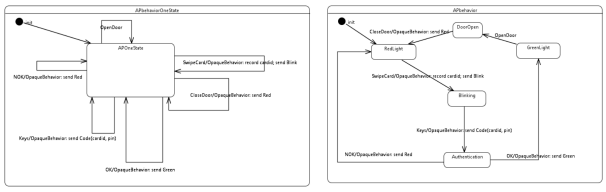


Exercise: Are these behaviors consistent?



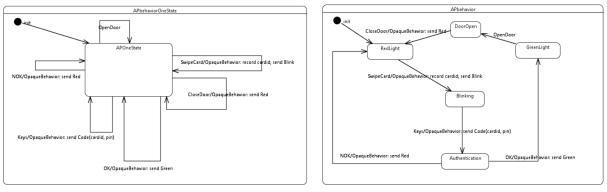


Exercise: Which state machine is the better description?



and why?

What if the user started keying the PIN at once?



APbehavior **may spot the problem**
APbehaviorOneState **will go on in error**

4.2.2 Why using different states?

- Several different states distinguishes between different situations
- In different situations, different reactions may be desirable to the same trigger
- A specific state represents in a compact way the whole history of behavior that led to reaching that state

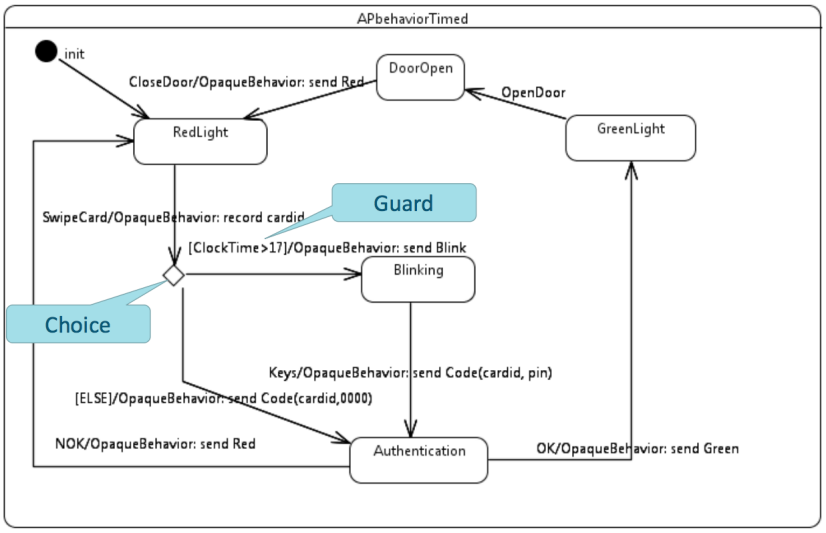
4.2.3 Guidelines and Reminders

- Even though the state machine was consistent with the sequence diagram, the state machine was flawed
  - The reason was that sequence diagrams are only partial descriptions of the whole, while state machines are complete descriptions of a part of the whole
- Use several states if you can
  - Each state representing a stable, recognizable situation

- We should supplement our state machine with all the possible different transitions
  - This would help us consider and handle most error situations

4.2.4 What if we need to modify a state machine?

- Our access control system should possibly be acting differently during working hours than at other times
- How well do state machines cope with modifications?



4.3 Summarizing

- State machines describe behavior of independently acting components
- Reactive systems are suitable for state machines
- Consistency checks between sequence diagrams and state machines are very useful (but not sufficient)
- State machines are robust in as much as additional functionality can often be included without ripple effects on other parts of the behavior

5 Lecture 5: Refinement I

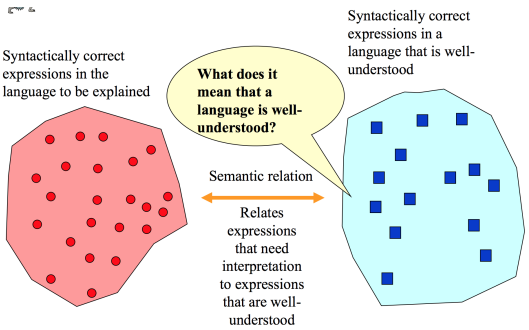
Objectives for the lectures on refinement

- Motivate the role of refinement
- Introduce and related the following notions of refinement
  - supplementing
  - narrowing
  - detailing
- Illustrate the use of these notions of refinement
  - the interplay between specification and refinement
- Illustrate the translation of theory into practice

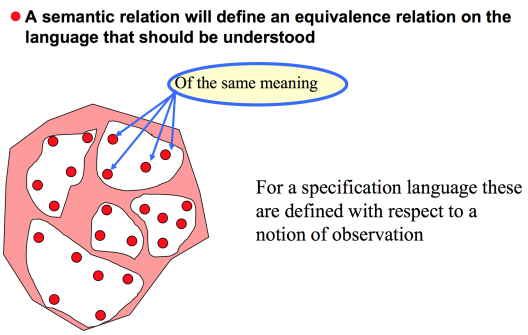
5.1 Three main concepts of language theory

- Syntax
  - The relationship between symbols or groups of symbols independent of content, usage and interpretation
- Semantics
  - The rules and conventions that are necessary to interpret and understand the content of language constructs
- Pragmatics
  - The study of the relationship between symbols or groups of symbols and their interpretation and usage

5.1.1 Semantic relation

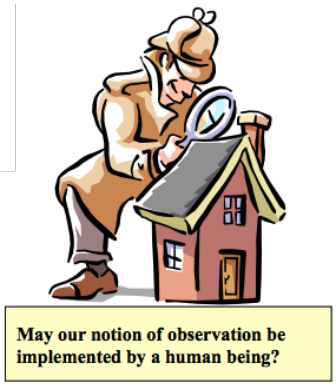


5.1.2 The need for a notion of observation



5.1.3 Definition of a notion of observation

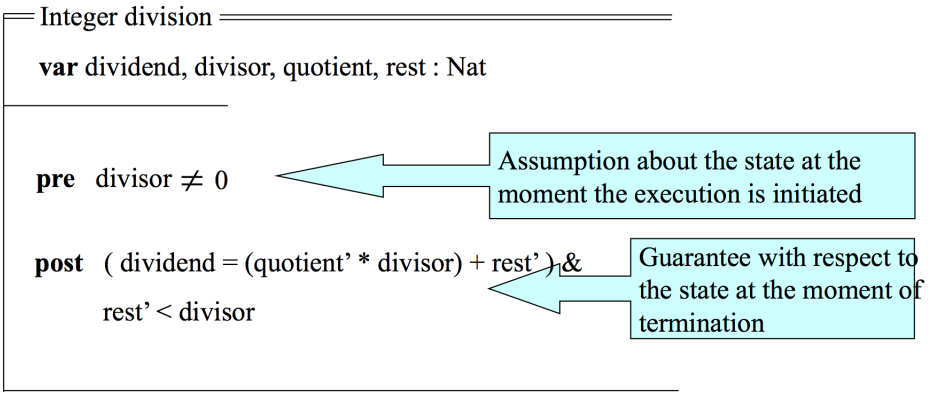
- May observe only external behavior
- May observe that nothing bad
- May observe that something eventually happens
- May observe any potential behavior
- May observe time with respect to a global clock



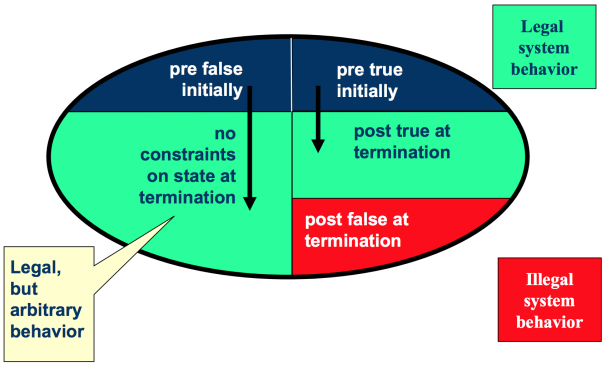
5.2 Pre-post specifications, The origins of refinement

5.2.1 Pre-post specifications

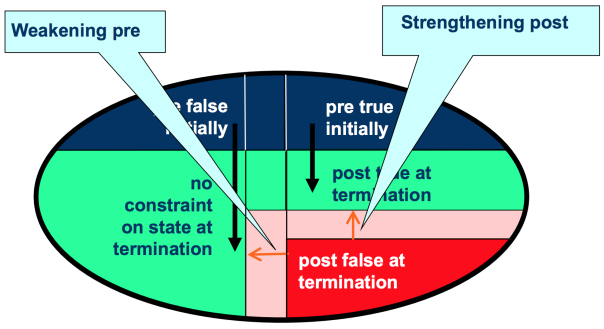
Pre-post specifications are based on the assumption-guarantee paradigm



5.2.2 Semantics of pre-post specifications



5.2.3 Refinement in pre-post



5.2.4 Weakening the pre-condition (assumption)

Integer division

var dividend, divisor, quotient, rest : Nat

pre **true**

post

if divisor ≠ 0 then

(dividend = (quotient' * divisor) + rest') & rest' < divisor

else quotient' = 0

5.2.5 Strengthening the post-condition (guarantee)

Integer division

var dividend, divisor, quotient, rest : Nat

pre divisor ≠ 0

post (dividend = (quotient' * divisor) + rest') &
rest' < divisor & dividend' = dividend &
divisor' = divisor

5.3 STAIRS - Refinement in UML

5.3.1 Motivation

- Exploit classical theory of refinement in a practical UML setting
  - From theory to practice, and not the other way around
- Sequence diagrams can be used to capture the meaning of other UML description techniques for behavior
- By defining refinement for sequence diagrams we therefore implicitly define refinement for UML

5.3.2 Traces for sequence diagrams summarized

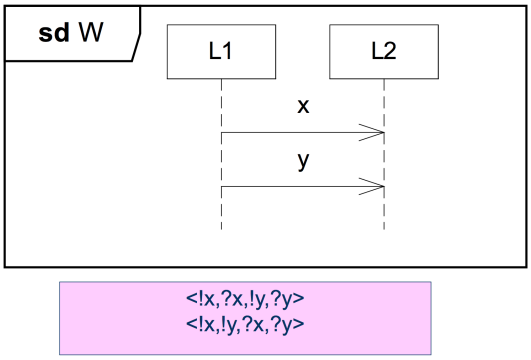
- Traces for sequence diagrams are sequences for events

  $< e1, e2, e3, e4, e4, e1, e2, e5, \ldots\ldots\ldots\ldots\ldots >$
- An event represent either the transmission or reception of messages
  - ?m - reception of message m
  - !m - transmission of message m
- Events are instantaneous
- A trace may be finite
  - termination, deadlock, infinite waiting, crash
- A trace may also be infinite
  - infinite loop, intended non termination

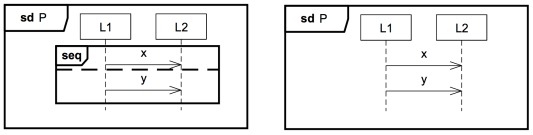
5.3.3 Causality and weak sequencing

- Causality:
  - a message can never be received before it has been transmitted
  - the transmission event for a message is therefore always ordered before the reception event for the same message
- Weak sequencing:
  - events from the same lifeline are ordered in the same order as on the lifeline

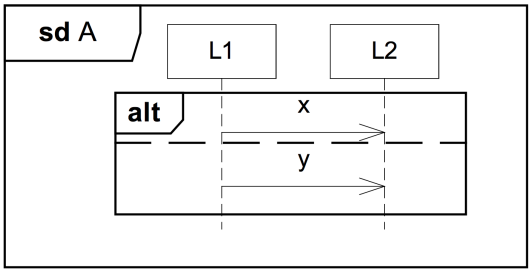
5.3.4 Weak sequencing



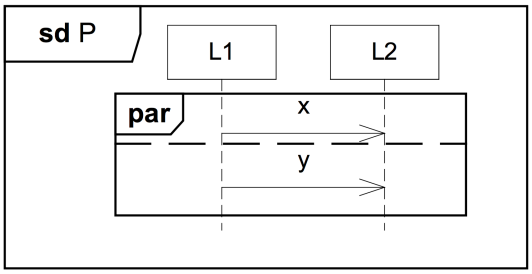
5.3.5 These two diagrams are semantically the same



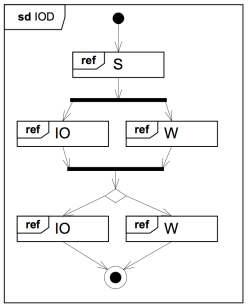
5.3.6 Alternative composition



5.3.7 Parallel composition

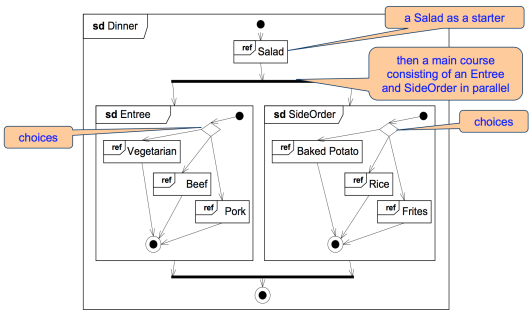


5.3.8 Interaction overview diagram

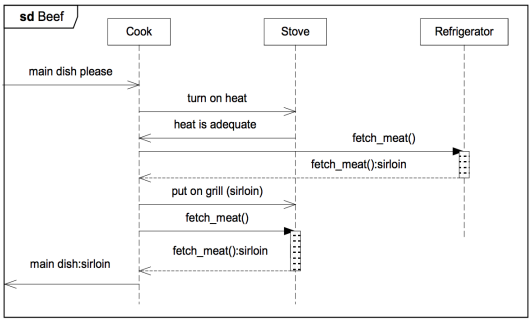


S **seq** (IO **par** W) **seq** (IO **alt** W)

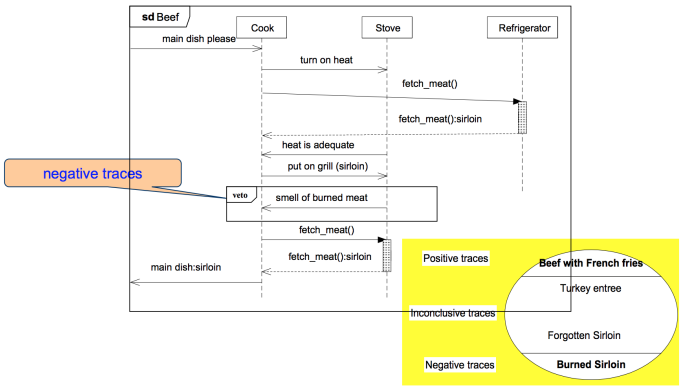
5.3.9 Dinner



5.3.10 Some potential positive traces of Beef



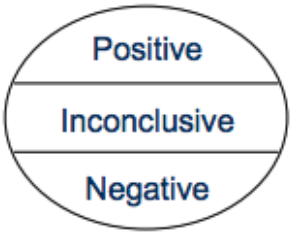
5.3.11 Potential negative Beef experiences



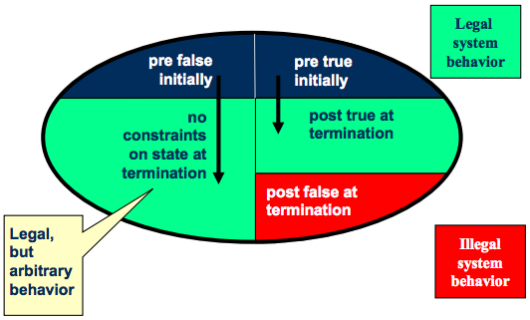
The execution does not stop even if the "veto" occurs, it finishes the execution

5.3.12 STAIRS semantics: simple case

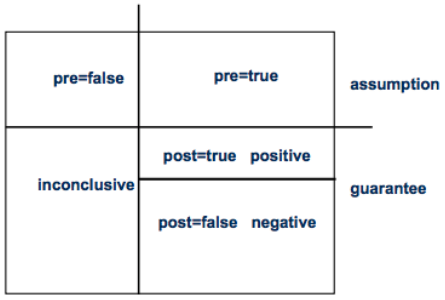
- Each positive execution is represented by a trace
- Each negative execution is represented by a trace
- The semantics of a sequence diagram is a pair of sets of traces (Positive, Negative)
- All other traces over the actual alphabet of events are inconclusive



5.3.13 Semantics of pre-post specifications

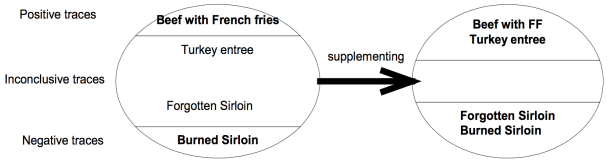


5.3.14 Comparing STAIRS with pre-post

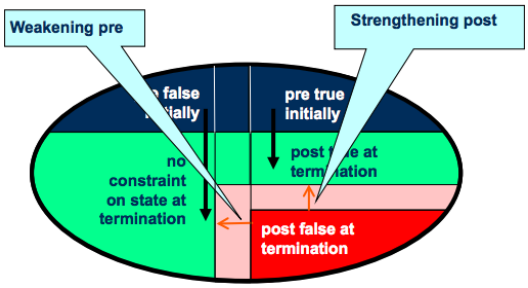


5.3.15 STAIRS: supplementing

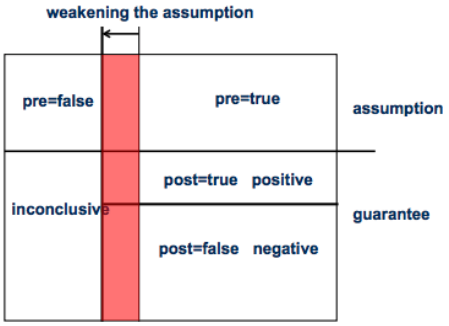
- Supplementing involves reducing the set of inconclusive traces by redefining inconclusive traces as either positive or negative
- Positive trace remains positive
- Negative trace remains negative



5.3.16 Refinement in pre-post

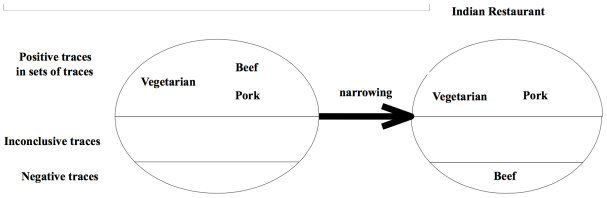


5.3.17 Supplementing in pre-post



5.3.18 STAIRS: narrowing

- Narrowing involves reducing the set of positive traces by redefining them as negative
- Inconclusive traces remain inconclusive
- Negative trace remains negative



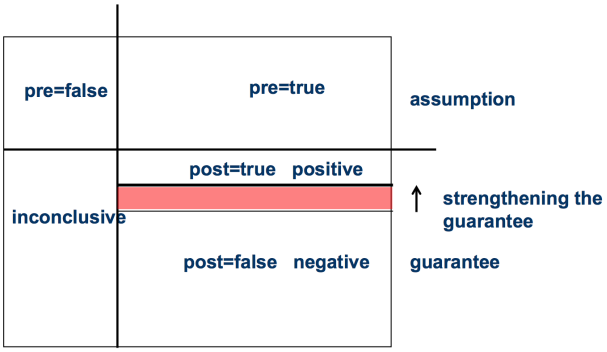
5.3.19 Indirect definition: Refinement in STAIRS

- A sequence diagram B is a general refinement of a sequence diagram A if
  - A and B are semantically identical
  - B can be obtained from A by supplementing
  - B can be obtained from A by narrowing
  - B can be obtained from A by a finite number of steps

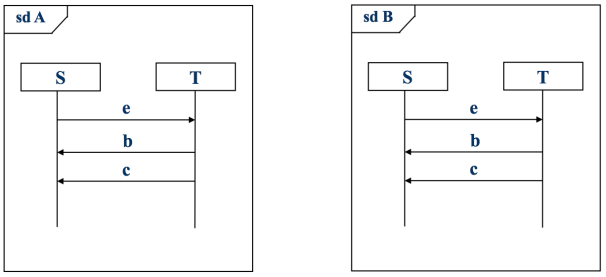
  $A -> C1 -> C2 -> \ldots . -> Cn -> B$

  each of which is either a supplementing or a narrowing

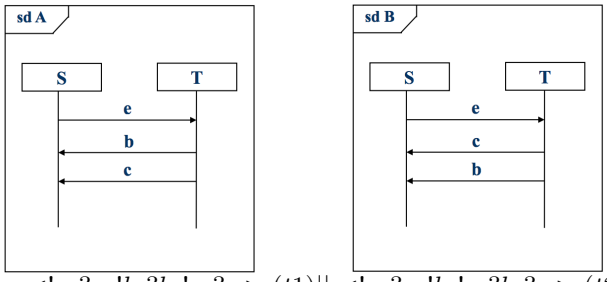
5.3.20 Narrowing in pre-post



Is B a refinement of A?

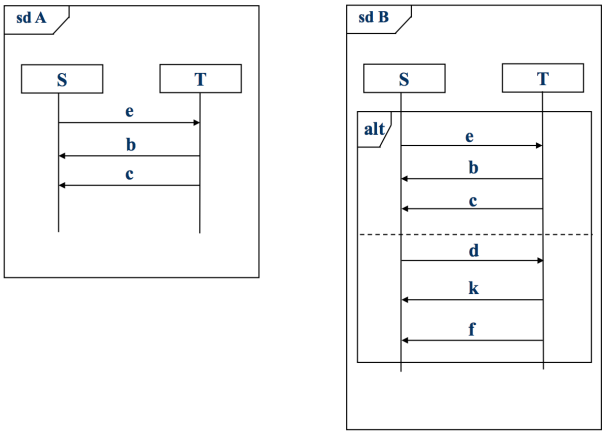


Is B a refinement of A?



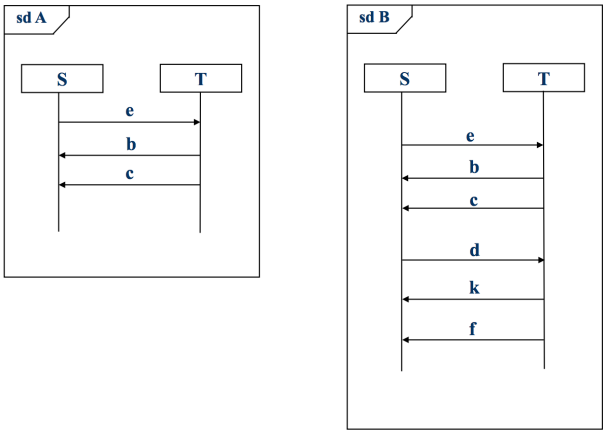
$A = <!e, ?e, !b, ?b, !c, ?c > (t1) || <!e, ?e, !b, !c, ?b, ?c > (t2)$
$B = <!e, ?e, !c, ?c, !b, ?b > (t4) || <!e, ?e, !c, !b, ?c, ?b > (t3)$
$[A] = (t1, t2,)$
$[B] = (t3, t4,)$

Is B a refinement of A?

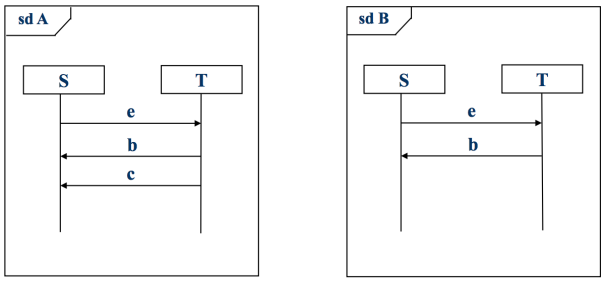


$[B'] = t1, t2, t5, t6$

Is B a refinement of A?



Is B a refinement of A?



6 Lecture 6: Refinement II

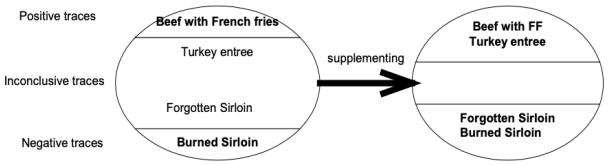
Outline

- Refinement summarized
- Inherent non-determinism (also called explicit non-determinism)

6.1 Refinement summarized

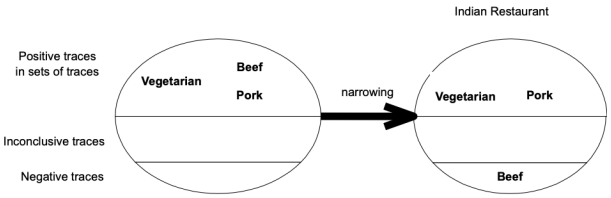
6.1.1 Supplementing

- Supplementing involves reducing the set of inconclusive traces by redefining inconclusive traces as either positive or negative
  - Positive trace remains positive
  - Negative trace remains negative



6.1.2 Narrowing

- Narrowing involves reducing the set of positive traces by redefining them as negative
  - Inconclusive traces remain inconclusive
  - Negative traces remain negative



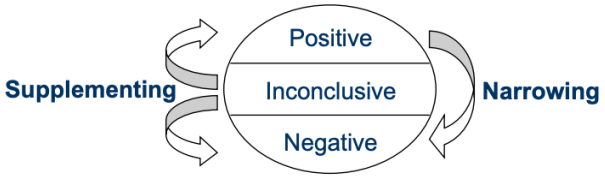
6.2 Direct definition of refinement

- A sequence diagram B is a refinement of a sequence diagram A if
  - every trace classified as negative by A is also classified as negative by B
  - every trace classified as positive by A is classified as either positive or negative by B

6.3 Refinement formalized

An interaction obligation o'=(p',n') is a refinement of an interaction obligation o=(p,n) iff (if and only if)

- n $\subseteq$ n'
- p $\subseteq$ p'Un'



6.4 Inherent non-determinism

6.4.1 Underspecification and inherent non-determinism

- Underspecification: Several alternative behaviours are considered equivalent (serve the same purpose)
- Inherent non-determinism: Alternative behaviours that must all be possible for the implementation
- These two should be described differently

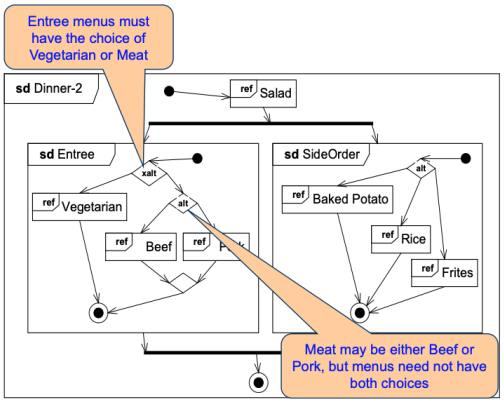
6.4.2 The need for both *alt* and *xalt*

- Potential non-determinism captured by alt allows abstraction and inessential non-determinism
  - Under-specification
  - Non-critical design decisions may be postponed
- Inherent or explicit non-determinism captured by xalt characterizes non-determinism that must be reflected in every correct implementation in one way or another.
  - Makes it possible to specify games
  - Important in relation to security
  - Also helpful as a means of abstraction

Example: an appointment system

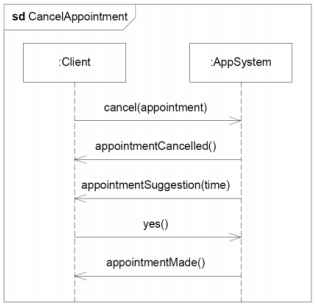
- A system for booking appointments used by e.g. dentists
- Functionality:
  - MakeAppointment: The client may ask for an appointment
  - CancelAppointment: The client may cancel an appointment
  - Payment: The system may send an invoice message asking the client to pay for the previous or an unused appointment.

Restaurant example with both alt and xalt



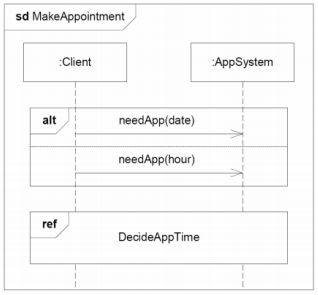
xalt vs alt (1): CancelAppointment

- This specification has two positive traces
- Whether reception of appointmentCancelled() occurs before or after sending of appointmentSuggestion(...) is not important
- Underspecification due to weak sequencing



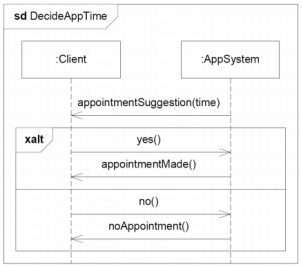
xalt vs alt (2): MakeAppointment

- May ask for either a specific date or a specific hour of the day (e.g. in the lunch break)
- The system is not required to offer both alternatives
- Underspecification expressed by the alt operator



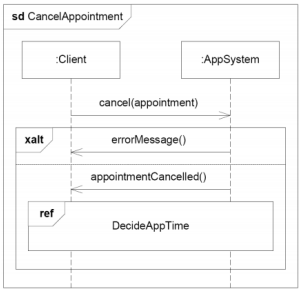
xalt vs alt (3): DecideAppTime

- The system must be able to handle *both* yes() and no() as reply messages from the client
- This is *not* underspecification
- Therefore the alternatives are expressed by the xalt operator



xalt vs alt (4): CancelAppointment

- The condition for choosing errorMessage() or appointmentCancelled() is not shown
- Both alternatives should be possible
- The choice is made by the system

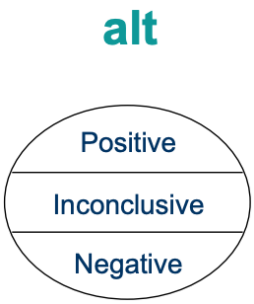


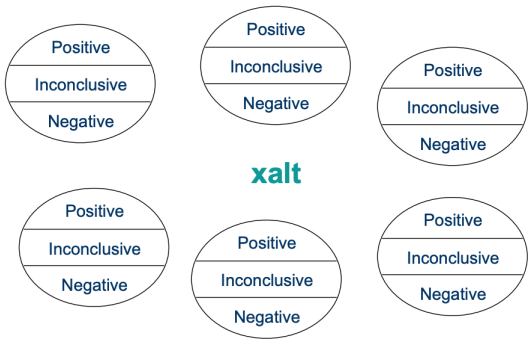
6.4.3 The pragmatics of alt vs xalt

- Use alt to specify alternatives that represent similar traces, i.e. to model (underspecification)
- Use xalt to specify alternatives that must all be present in an implementation, i.e. to model
  - inherent nondeterminism, as in the specification of a coin toss
  - alternative traces due to different inputs that the system must be able to handle (as in DecideAppTime)
  - alternative traces where the conditions for these being positive are abstracted away (as in CancelAppointment on slide 12)

6.4.4 Semantics - general case

- The semantics of a sequence diagram *without occurrences* of **xalt** is a single interaction obligation (p, n)
- The semantics of a sequence diagram *with occurrences* of **xalt** is a set of arbitrarily many interaction obligations $(p1, n1), (p2, n2), ..., (pK, nK)$





6.4.5 Notational convention

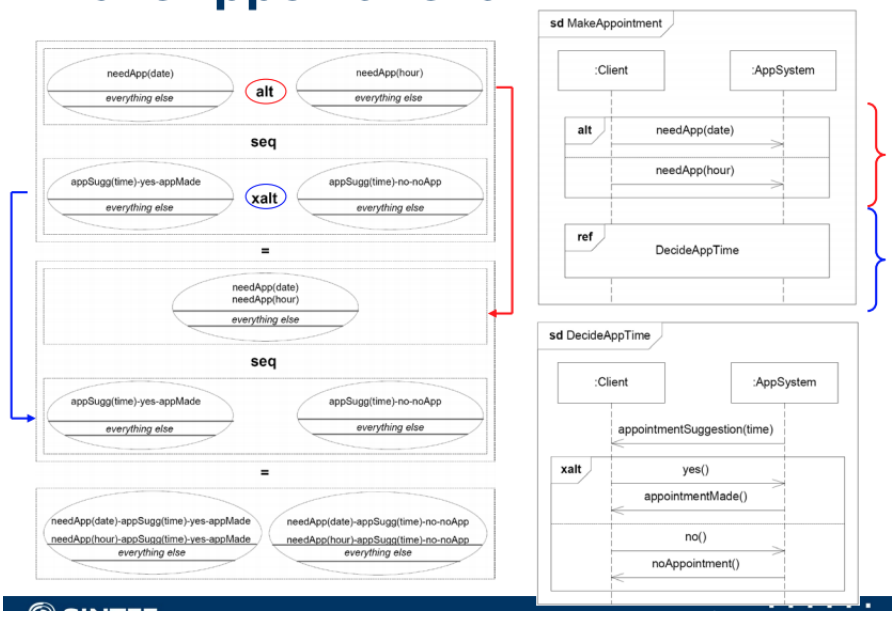
- For any sequence diagram d, [[d]] denotes its sematics
- We may think of [[ ]] as a function of the following type

[ ]: SequenceDiagram → Set of InteractionObligation

Formal semantics of alt and xalt

- Alt combines interaction obligations:
  - $[[d_1 \text{ alt } d_2]] \stackrel{\text{def}}{=} \{o_1 \uplus o_2 \mid o_1 \in [[d_1]] \wedge o_2 \in [[d_2]]\}$
    - Inner union of interaction obligations $\uplus$:
    - $(p_1, n_1) \uplus (p_2, n_2) \stackrel{\text{def}}{=} (p_1 \cup p_2, n_1 \cup n_2)$
- Xalt results in distinct interaction obligations:
  - $[[d_1 \text{ xalt } d_2]] \stackrel{\text{def}}{=} [[d_1]] \cup [[d_2]]$

Informal illustration of MakeAppointment



7 Lecture 7: Refinement III

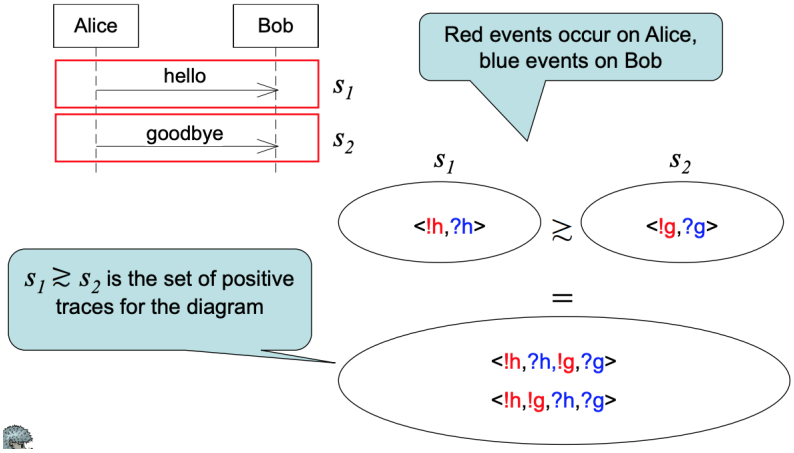
Outline

- Weak sequencing
- Give guidelines on
  - the use of operators (pragmatics of creating interactions)
    - alt versus xalt
    - specifying negative behaviour (refuse, veto, assert)
    - seq
  - refinement (pragmatics of refining interactions)

7.1 Weak sequencing

- Combine interaction fragments by seq
- Definition of weak sequencing of trace sets:
- $s_1 \gtrsim s_2$ denotes the set of all traces that may be constructed by selecting one trace t_1 from s_1 and one trace t_2 from s_2 and combining them in such a way that for each lifeline, the events from t_1 comes before the events from t_2
- Note: if s_1 or s_2 is empty then $s_1 \gtrsim s_2$ is also empty
- Remember: if the message hello is sent from l_1 to l_2, then the event !hello occurs on l_1 and ?hello occurs on l_2

7.1.1 Weak sequencing of trace sets

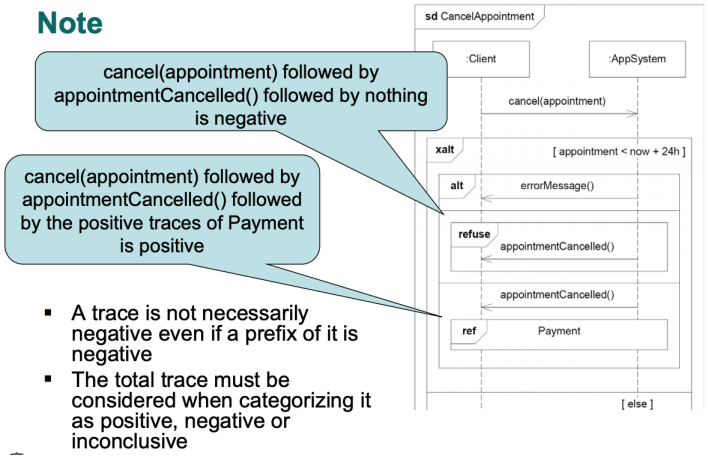


7.1.2 Weak sequencing of interaction obligations

- $(p_1,n_1)\gtrsim(p_2,n_2) \stackrel{\text{def}}{=} (p_1\gtrsim p_2,\ (n_1\gtrsim p_2)\cup(n_1\gtrsim n_2)\cup(p_1\gtrsim n_2))$
- Traces composed exclusively by positive traces become positive
- Traces composed with at least one negative trace become negative

7.1.3 Formal semtics of seq

- $[[d_1 \text{ seq } d_2]] \stackrel{\text{def}}{=} \{o_1\gtrsim o_2 \mid o_1\in[[d_1]]\wedge o_2\in[[d_2]]\}$
- seq is the implicit composition operator
- o_i is shorthand for $(p_i,\ n_i)$
- Note: For better readability we give the binary versions of the operators in this presentation. N-ary version are used in the paper.



7.1.4 The pragmatics of weak sequencing

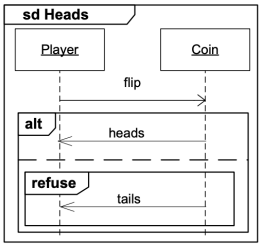
- Be aware that by weak sequencing
  - a positive sub-trace followed by a positive sub-trace is positive
  - a positive sub-trace followed by a negative sub-trace is negative
  - a negative sub-trace followed by a positive sub-trace is negative
  - a negative sub-trace followed by a negative sub-trace is negative
  - the remaining trace combinations are inconclusive
- Remember the definition:
  $(p_1,n_1)\gtrsim(p_2,n_2) \stackrel{\text{def}}{=} (p_1\gtrsim p_2,\ (n_1\gtrsim p_2)\cup(n_1\gtrsim n_2)\cup(p_1\gtrsim n_2))$

7.1.5 opt and skip

- [[opt d]] $\stackrel{\text{def}}{=}$ [[skip alt d]]
- [[skip]] $\stackrel{\text{def}}{=}$ {({<>},∅)}
  - A single interaction obligation where only the empty trace <> is positive and the set of negative traces is empty

Specifying negative behaviour: refuse

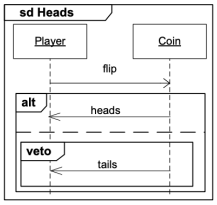
- [[refuse d]] $\stackrel{\text{def}}{=}$ {({},$p \cup n$) | (p,n)∈[[d]]}
- All interaction obligations in [[refuse d]] have empty positive sets
- This means that all interaction obligations in [[d_1 seq (refuse d_2)]] have empty positive sets
  - and the same applies to [[(refuse d_1) seq d_2]]



- [[Heads]] = {({<!f, ?f, !h, ?h>}, {<!f, ?f, !t, ?t>})}

Specifying negative behaviour: veto

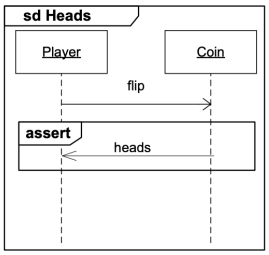
- [[veto d]] $\stackrel{\text{def}}{=}$ [[skip alt (refuse d)]]
- ... which means that [[veto d]] = {({<>},$p \cup n$) | ($p \cup n$)∈[[d]]}



- [[Heads]] = {({<!f, ?f, !h, ?h>, <!f, ?f>} , {<!f, ?f, !t, ?t>})}

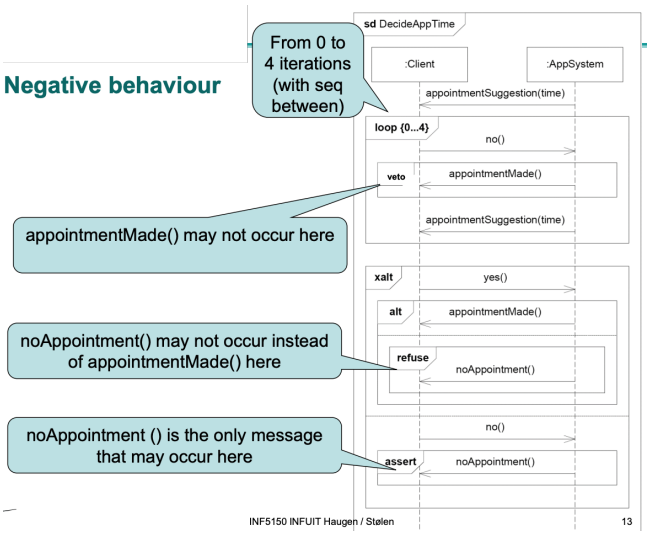
Specifying negative behaviour : assert

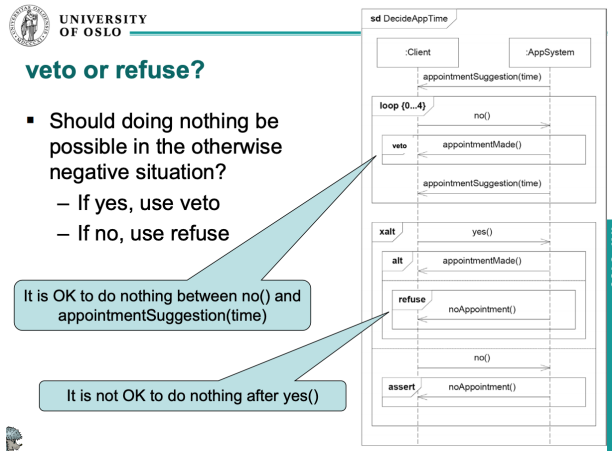
- By using assert, all inconclusive traces are redefined as negative
- This ensures that for each interaction obligation, at least one of its positive traces will be implemented in the final implementation
- [[assert d]] $\stackrel{\text{def}}{=}$ {(p,$n \cup (\mathcal{H} \setminus p)$) | ($p$,$n$)∈[[$d$]]}

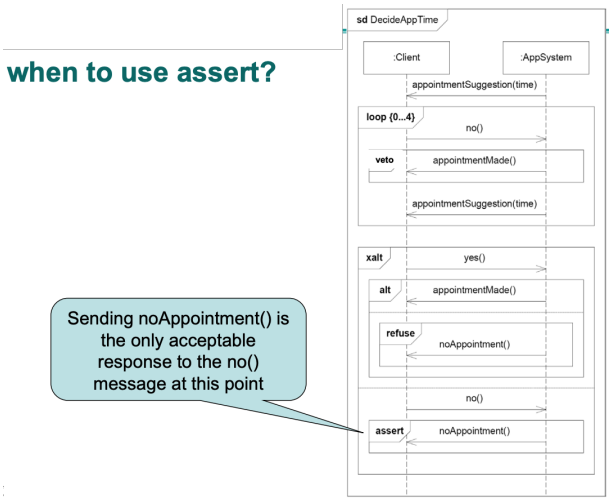


- [[Heads]] = {({<!f, ?f, !h, ?h>}, n)}
- n = all traces where the first event on the lifeline of Player is !f and the first event on the lifeline of Coin is ?f except the trace <!f, ?f, !h, ?h>

Negative behaviour







7.1.6 The pragmatics of negation

- To effectively constrain the implementation, the specification should include a reasonable set of negative traces
- Use refuse when specifying that one of the alternatives in an alt-construct represents negative traces
- Use veto when the empty trace (i.e. doing nothing) should be positive, as when specifying a negative message in an otherwise positive scenario
- Use assert on an interaction fragment when all positive traces for that fragment have been described (Use assert with caution!)

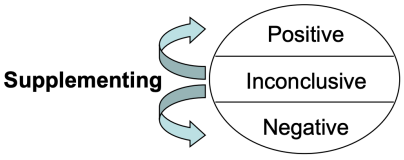
7.2 The pragmatics of refining interactions

7.2.1 The use of supplementing

- Inconclusive trace are recategorized as either positive or negative (for an interaction obligation)
- New situations are considered
  - adding fault tolerance
  - new user requirements
  - ...
- Typically used in early phases

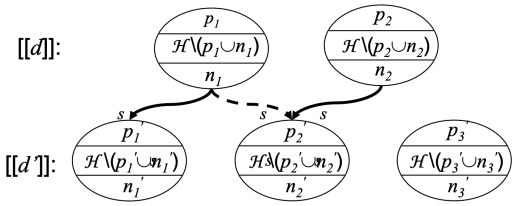
7.2.2 Supplementing of interaction obligations

- $(p,n) \rightsquigarrow_s (p',n') \stackrel{\text{def}}{=} p \subseteq p' \wedge n \subseteq n'$



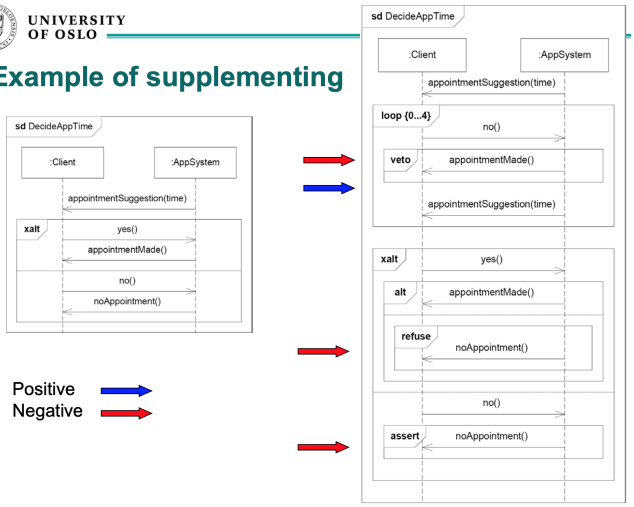
Supplementing of specifications

- $d \rightsquigarrow_s d' \stackrel{\text{def}}{=} \forall o \in [[d]]: \exists o' \in [[d']]: o \rightsquigarrow_s o'$
- d' is a supplementing of d if
  - for every interaction obligation o in [[d]] there is at least one interaction obligation o' in [[d']] such that o' is a supplementing of o



UNIVERSITY OF OSLO

Example of supplementing

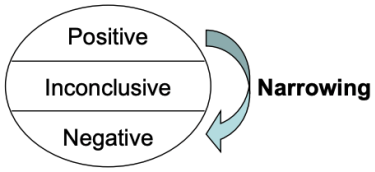


7.2.3 The pragmatics of supplementing

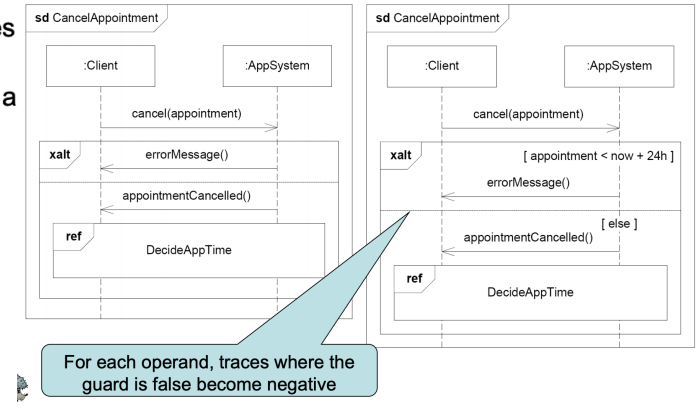
- Use supplementing to add positive or negative traces to the specification
- When supplementing, all of the original positive traces must remain positive, and all the original negative traces must remain negative
- Do not use supplementing on the operand of an assert (no traces are inconclusive in the operand)

Narrowing

- Reduce underspecification by redefining positive traces as negative
- For example adding guards, or replacing a guard with a stronger one
  - traces where the guard is false become negative
- $(p,n) \rightsquigarrow_n (p',n') \stackrel{\text{def}}{=} p' \subseteq p \wedge n' = n \cup (p \setminus p')$
- $d \rightsquigarrow_n d' \stackrel{\text{def}}{=} \forall o \in [[d]]: \exists o' \in [[d']]: o \rightsquigarrow_n o'$



Example of narrowing

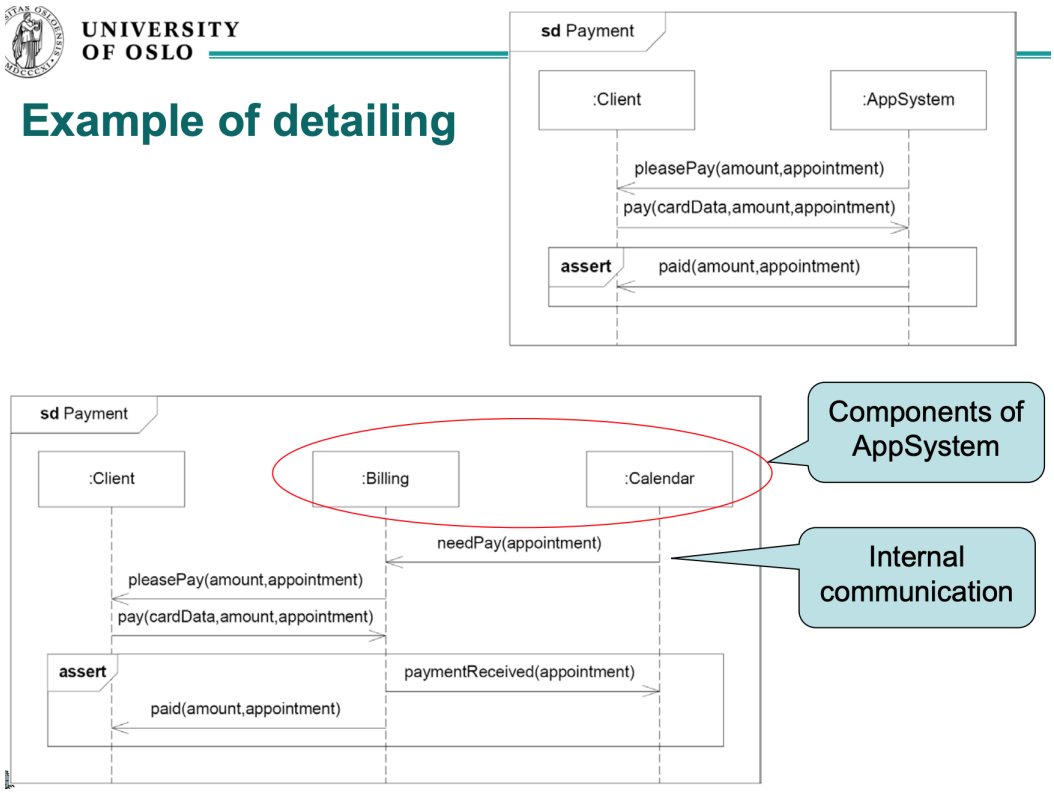


7.2.4 The pragmatics of narrowing

- Use narrowing to remove underspecification by redefining positive traces as negative
- In cases of narrowing, all of the original negative traces must remain negative
- Guards may be added to an alt-construct as a legal narrowing step
- Guards may be added to an xalt-construct as a legal narrowing step
- Guards may be narrowed, i.e. the refined condition must imply the original one

7.2.5 The use of detailing

- Reducing the level of abstraction by structural decomposition (One or more lifelines are decomposed)
- The positive and the negative traces are the same, except that
  - internal communication is hidden at the abstract level
  - events occurring on a composed lifeline at the abstract level occur instead on one of the sub-component lifelines



7.2.6 The pragmatics of detailing

- Use detailing to increase the level of granularity of the specification by decomposing lifelines
- When detailing, document the decomposition by creating a mapping L from the concrete to the abstract lifelines
- When detailing, make sure that the refined traces are equal to the original ones when abstracting away internal communication and taking the lifeline mapping to account

7.2.7 The use of general refinement

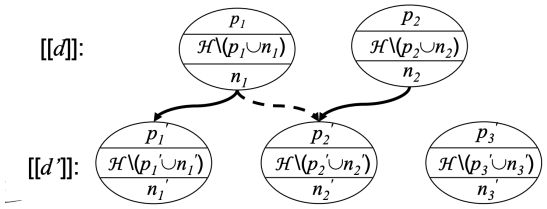
- A combination of supplementing, narrowing and detailing (not necessarily all three)
- Allows all positive traces to become negative, while previously inconclusive traces become positive
- To ensure that a trace *must* be present in the final implementation we need an interaction obligation where all other traces are negative

7.2.8 The pragmatics of general refinement

- Use general refinement to perform a combination of supplementing, narrowing and detailing in a single step
- To define that a particular trace *must* be present in an implementation use xalt and assert to characterize an obligation with this trace as the only positive one and all other traces as negative

General refinement (of sets of interaction obligations)

- $d \rightsquigarrow d' \stackrel{\text{def}}{=} \forall o \in [[d]] : \exists o' \in [[d']] : o \rightsquigarrow o'$
- d' is a general refinement of d if
  - for every interaction obligation o in $[[d]]$ there is at least one interaction obligation o' in $[[d']]$ such that o' is a general refinement of o
- New interaction obligations may also be added
  - that do not refine any obligation at the abstract level



Compositionality

- A refinement operator $\rightsquigarrow$ is compositional if it is
  - reflexive: $d \rightsquigarrow d$
  - transitive: $d \rightsquigarrow d' \wedge d' \rightsquigarrow d'' \Rightarrow d \rightsquigarrow d''$
  - the operators refuse, veto, alt, xalt and seq are monotonic w.r.t. $\rightsquigarrow$:
    - $d \rightsquigarrow d' \Rightarrow$ refuse $d \rightsquigarrow$ refuse d'
    - $d \rightsquigarrow d' \Rightarrow$ veto $d \rightsquigarrow$ veto d'
    - $d_1 \rightsquigarrow d_1' \wedge d_2 \rightsquigarrow d_2' \Rightarrow d_1$ alt $d_2 \rightsquigarrow d_1'$ alt d_2'
    - $d_1 \rightsquigarrow d_1' \wedge d_2 \rightsquigarrow d_2' \Rightarrow d_1$ xalt $d_2 \rightsquigarrow d_1'$ xalt d_2'
    - $d_1 \rightsquigarrow d_1' \wedge d_2 \rightsquigarrow d_2' \Rightarrow d_1$ seq $d_2 \rightsquigarrow d_1'$ seq d_2'
- Transitivity allows stepwise development
- Monotonicity allow different parts of the specification to be refined separately
- Supplementing, narrowing, detailing, general refinement and limited refinement are all compositional ☺

8 Lecture 8: Security Risk Assessment I

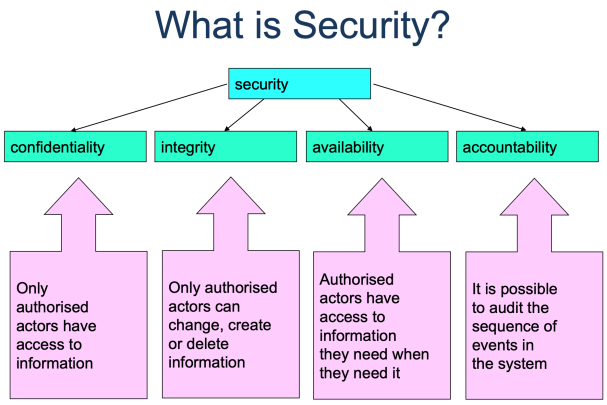
Lecture overview

- What is security?
- What is risk?
- What is risk management?
- What is the relationship to cyber security?
- What is CORAS?

8.0.1 What is Security Risk Assessment?

Security risk assessment is a specialized form of risk assessment focusing on security risks

8.1 What is Security?



8.1.1 Security is more than Technology

- What good is security if no one can use the systems?
- Requires more than technical understanding
- Incidents often of non-technical origin
- Requires uniform description of the whole
  - how it is used, the surrounding organisation, etc.

8.1.2 Security should not be an "afterthought"

- Security issues solved in isolation
- Costly redesign
- Security not completely integrated

8.2 What is Risk?

Two categories of risk assessment: offensive and defensive risk assessment

Many kinds of risk

- Contractual risk
- Economic risk
- Operational risk
- Environmental risk
- Health risk
- Political risk
- Legal risk
- Security risk

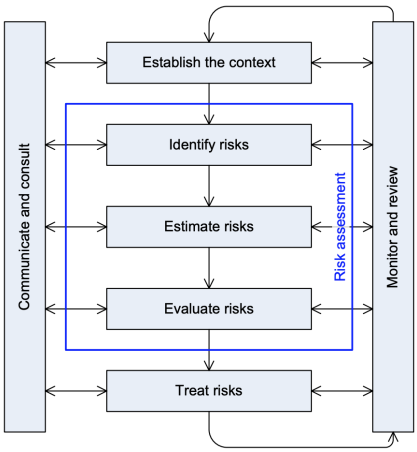
8.2.1 Definition of Risk from ISO 31000

Risk: Effect of uncertainty on objectives

- NOTE 1 An effect is a deviation from the expected — positive and/or negative
- NOTE 2 Objectives can have different aspects (such as financial, health and safety, and environmental goals) and can apply at different levels (such as strategic, organization-wide, project, product and process)
- NOTE 3 Risk is often characterized by reference to potential events and consequences, or a combination of these
- NOTE 4 Risk is often expressed in terms of a combination of the consequences of an event (including changes in circumstances) and the associated likelihood of occurrence
- NOTE 5 Uncertainty is the state, even partial, of deficiency of information related to, understanding or knowledge of an event, its consequence, or likelihood

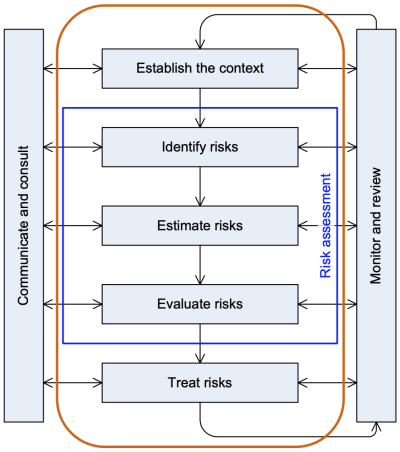
8.3 What is Risk Management?

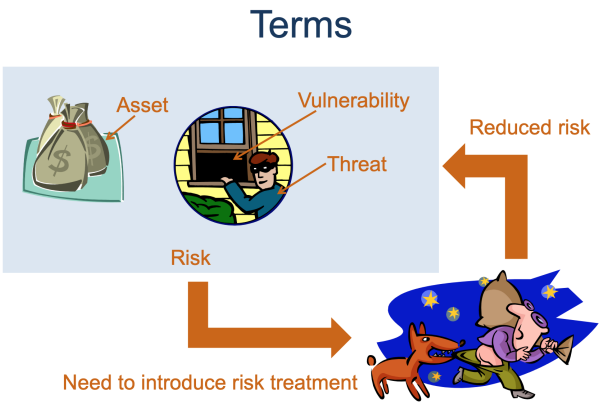
Risk management: Coordinated activities to direct and control an organization with regard to **risk**

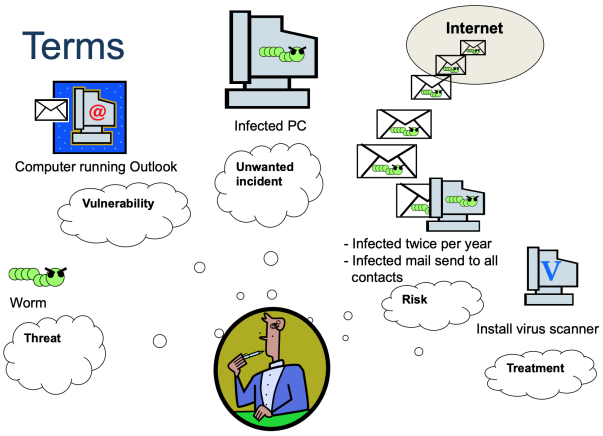


8.3.1 Risk Assessment involves

- Determining what can happen, why and how
- Systematic use of available information to determine the level of risk
- Prioritization by comparing the level of risk against predetermined criteria
- Selection and implementation of appropriate options for dealing with risk







8.4 Cyberspace, Cybersecurity and Cyber-risk

What is new with "cyber"?

Cyberspace: A *cyberspace* is a collection of interconnected computerized networks, including services, computer systems, embedded processors and controllers, as well as information in storage or transit.

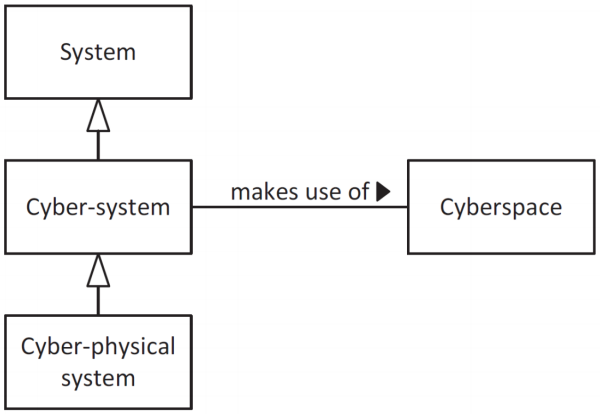
Cyber-system: A *cyber-system* is a system that makes use of a cyberspace

Cyber-physical system: A *cyber-physical system* is a cyber-system that controls and responds to physical entities through actuators and sensors.

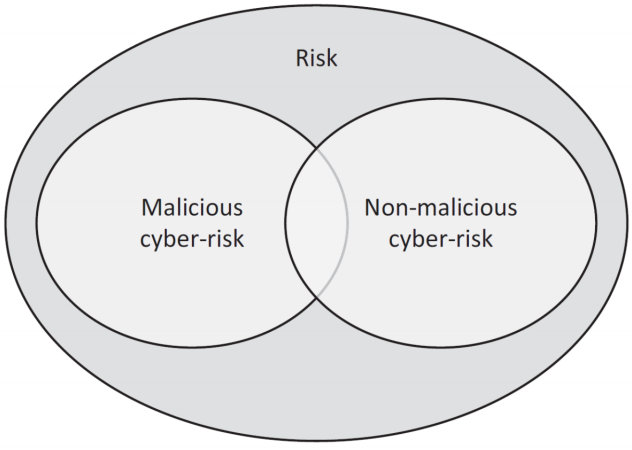
Cybersecurity: *Cybersecurity* is the protection of cyber-systems against cyber-threats

Cyber-threat: A *cyber-threat* is a threat that exploits a cyberspace

Cyber-risk: A cyber-risk is a risk that is caused by a cyber-threat



8.4.1 Summary



9 Security Risk Asessment Using CORAS

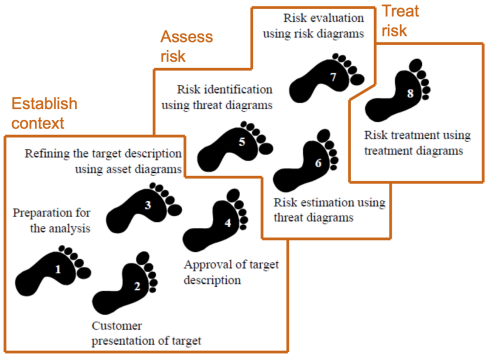
Overview

- What is CORAS?
- Main concepts
- Process of eight steps
- Risk modeling
- Semantics
- Calculus
- Tool support
- Further reading

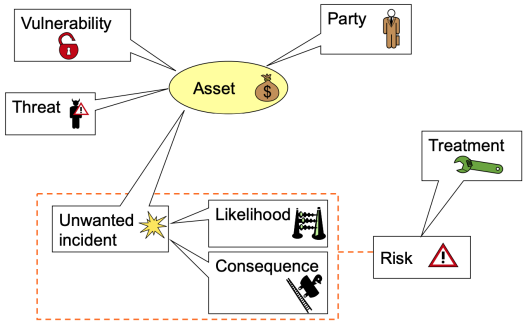
9.0.1 The CORAS method

- Asset-driven defensive risk analysis method
- Operationalization of ISO 31000 and ISO 27005 risk analysis process in 8 steps
- Detailed guidelines explaining how to conduct each step in practice
- Modeling guidelines for how to use the CORAS language

9.0.2 The 8 Steps of the CORAS Method



9.1 Main Concepts



9.1.1 Definitions

- **Asset**: Something to which a party assigns value and hence for which the party requires protection
- **Consequence**: The impact of an unwanted incident on an asset in terms of harm or reduced asset value
- **Likelihood**: The frequency or probability of something to occur
- **Party**: An organization, company, person, group or other body on whose behalf a risk analysis is conducted
- **Risk**: The likelihood of an unwanted incident and its consequence for a specific asset

- **Risk level**: The level or value of a risk as derived from its likelihood and consequence
- **Threat**: A potential cause of an unwanted incident
- **Treatment**: An appropriate measure to reduce risk level
- **Unwanted incident**: An event that harms or reduces the value of an asset
- **Vulnerability**: A weakness, flaw or deficiency that opens for, or may be exploited by, a threat to cause harm to or reduce the value of an asset

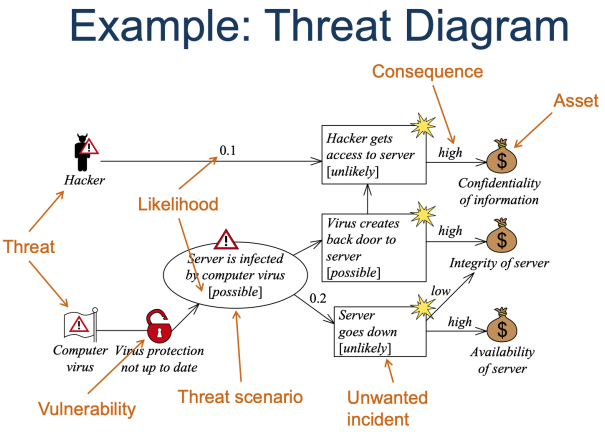
9.2 Risk Modeling

The CORAS language consists of five kinds of diagrams

- Asset diagrams
- Threat diagrams
- Risk diagrams
- Treatment diagrams
- Treatment overview diagrams

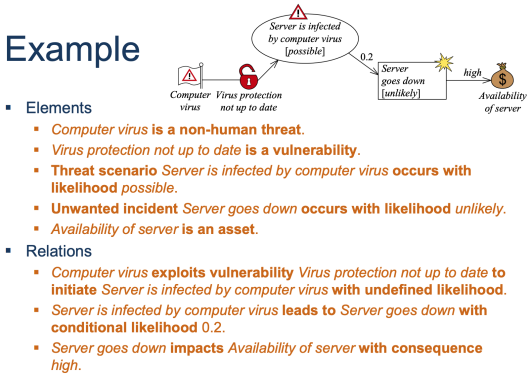
Each kind supports concrete steps in the risk analysis process
In addition there are three kinds of diagrams for specific needs

- High-level CORAS diagrams
- Dependent CORAS diagrams
- Legal CORAS diagrams



9.3 Semantics

- How to interpret and understand a CORAS diagram?
- Users need a precise and unambiguous explanation of the meaning of a given diagram
- Natural language semantics
  - CORAS comes with rules for systematic translation of any diagram into sentences in English
- Formal semantics



9.3.1 Criticism from System Developers

The CORAS language is too simplistic
It is too cumbersome to use graphical icons

9.3.2 Criticism from Risk Analysts

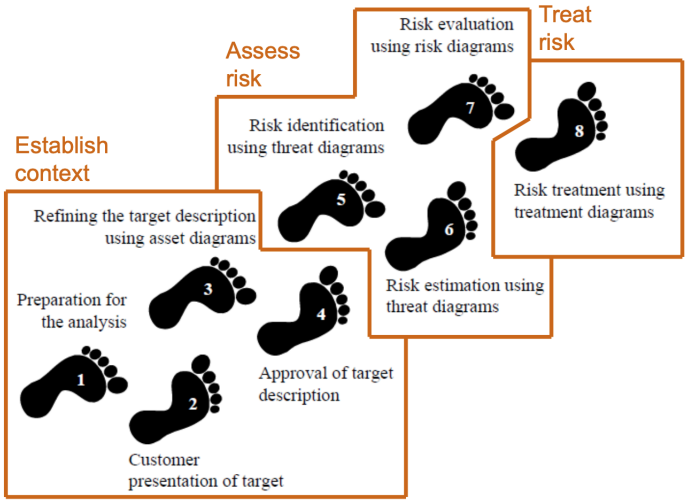
What's new with the CORAS language?
We have been using something similar for years, namely VISIO!

10 Lecture 9: Security Risk Assessment II

Lecture overview

- CORAS exeplified
- Walkthrough of the 8 steps based on the ATM example
- Calculation of frequencies

10.1 The 8 steps of the CORAS method



10.1.1 Step 1: Preparation for the assessment

Objectives

- Obtain information about the customer, purpose and domain of assessment
- Decide size of assessment
- Ensure customer is prepared
- Practical organization of analysis

Interaction between the customer and the analysis team

- By mail, phone or face-to-face

Preliminaries

- Customer is a national air navigation service provider
- The customer decides on an assessment of 250 person-hours

Target of risk assessment

- The role of the Air Traffic Controllers (ATCOs) in the process of arrival management
- Information provisioning
- Compliance (respect to privacy, air traffic, national laws etc.)

Air Traffic Control (ATC)

- Maintain horizontal and vertical separation among aircrafts and possible obstacles
- Limited interaction with the external world
- Humans at the centre of decisions and work process

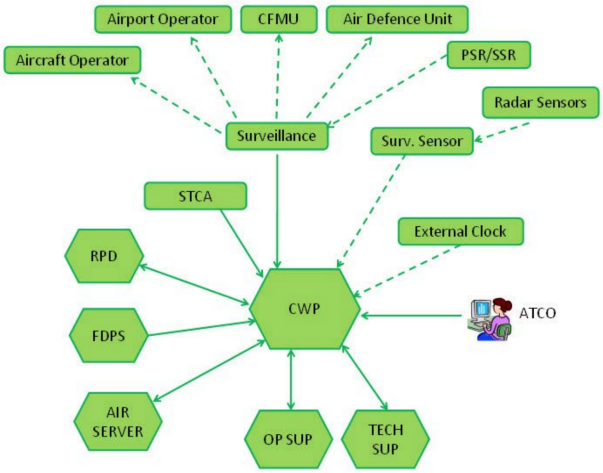
10.1.2 Step 2: Customer presentation of target

Objective

- Obtain understanding of what to assess
- Identify focus, scope and assumptions

Face-to-face between the customer and the assessment team

- Present CORAS terminology and method
- Collect as much information as possible



10.1.3 Step 3: Refine target description using asset diagrams

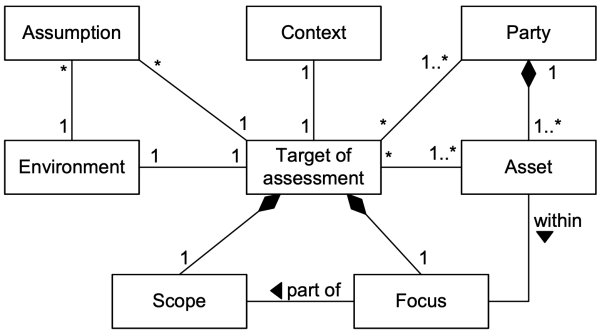
Objective

- Ensure common understanding of target including scope, focus and assets

Face-to-face meeting

- Assessment team presents their understanding of the target
- Assets are identified
- High-level assessment

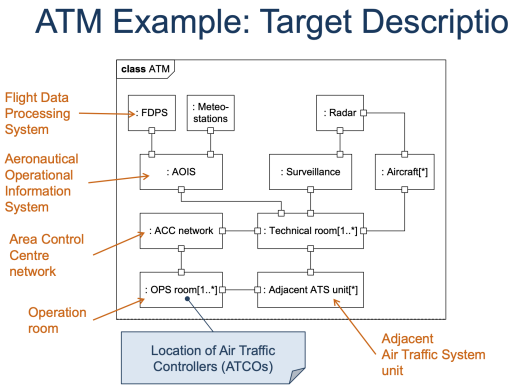
Target Description



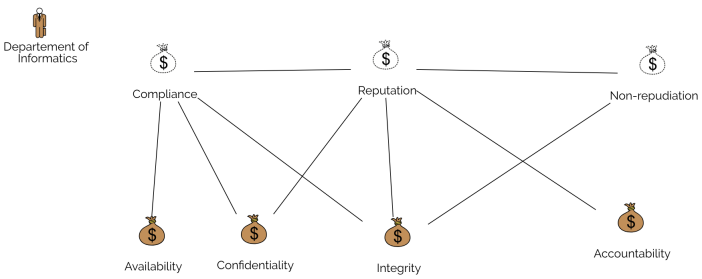
- **Asset**: Something to which a party assigns value and hence for which the party requires protection
- **Assumption**: Something we take as granted or accept as true (although it may not be so)
- **Context of assessment**: The premises for and the background of a risk assessment, including its purposes
- **Environment of target**: The surrounding things of relevance that may affect or interact with the target; in the most general case, the rest of the world
- **Focus of assessment**: The main issue or central area of attention in a risk assessment
- **Party**: An organization, company, person, group or other body on whose behalf a risk assessment is conducted
- **Scope of assessment**: The extent or range of a risk assessment. The scope defines the border of the assessment, in other words what is held inside of and what is held outside of the assessment
- **Target of assessment**: The system, organization, enterprise, or the like that is the subject of a risk assessment

ATM target description

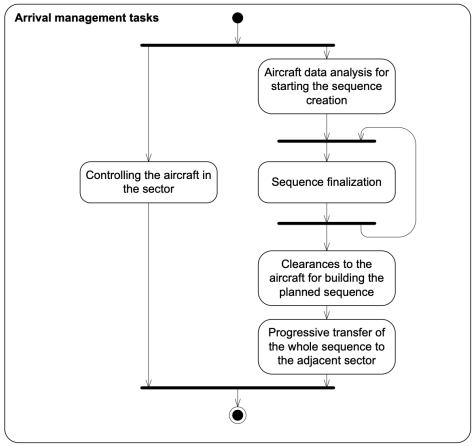
- Conceptual overview using UML class diagrams
- Activities using UML structured classifier and activity diagrams



Direct and indirect asset example

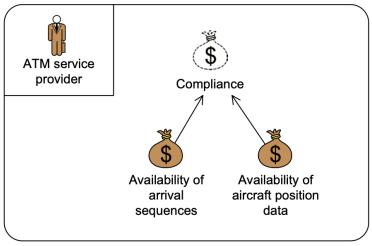


ATM Example: Target Description



ATM example: Asset identification

- Assets are the values the parties of the analysis wants to protect
- Identified assets are presented in CORAS asset diagrams



ATM Example: High-level analysis

- Threat, vulnerabilities, threat scenarios and unwanted incidents are identified in a brainstorming session
- Identify biggest worries and increase understanding of focus and scope

Who/what causes it?	How? What is the scenario or incident? What is harmed	What makes it possible?
Component failure; power loss	Provisioning of information to ATCO fails due to loss of CWP (Controller Working Position)	Insufficient CWP maintenance
Software error	The consolidation of data from several radar sources fails	Lack of redundant aircraft tracking systems
Component failure; radar disturbance	Malfunctioning of radar antenna; loss of aircraft tracking	Insufficient radar maintenance
Software bugs	False or redundant alerts from safety tool	Insufficient software testing

10.1.4 Step 4: Approval of Target Description

Objective

- Ensure target description is correct and complete
- Ranking of assets
- Scales for risk estimation
- Risk evaluation criteria

Face-to-face meeting

- Structured walk-through of target description
- Plenary discussion of assets, scales and criteria

Consequence Scales

- One consequence scale for each asset is defined
  - Note: Sometimes one scale applies to several assets
- Consequences can be qualitative or quantitative
- Scales can be continuous, discrete or with intervals

ATM Example: Consequence scale

The same consequence scale applies to the two direct availability assets

Consequence	Description
Catastrophic	Catastrophic accident
Major	Abrupt maneuver required
Moderate	Recovery from large reduction in separation
Minor	Increasing workload of ATCOs or pilots
Insignificant	No hazardous effect on operations

The consequence and likelihood scales are partly based on requirements and advisory material provided by EUROCONTROL

Consequence Scale example for privacy

Consequence	Description
Completely compromised	All private information leaked and disptributed to everyone
Catastrophic	All private information leaked to one entity
Major	Identify theft
Moderate	personal information lost
Minor	Basic information lost
Insignificant	no information lost

Likelihood Scales

- One likelihood scale is defined
  - The scale is used for all unwanted incidents and threat scenarios
- Likelihoods can be
  - Qualitative or quantitative
  - Probabilities or frequencies
- Scales can be continuous, discrete or with intervals

ATM Example: Likelihood Scale

Likelihood	Description
Certain	A very high number of similar occurrences already on record; has occurred a very high number of times at the same location/time
Likely	A significant number of similar occurrences already on record; has occurred a significant number of times at the same location
Possible	Several similar occurrences on record; has occurred more than once at the same location
Unlikely	Only very few similar incidents on record when considering a large traffic volume or no records on a small traffic volume
Rare	Has never occurred yet throughout the total lifetime of the system

ATM Example: Risk Evaluation Criteria

Likelihood \ Consequence	Insignificant	Minor	Moderate	Major	Catastrophic
Rare					
Unlikely					
Possible					
Likely					
Certain					

- **High risk:** Unacceptable and must be treated
- **Medium risk:** Must be evaluated for possible treatment
- **Low risk:** Must be monitored

Qualitative scale types:

- Nominal Scale
- Ordinal Scale
- Interval Scale
- Ratio Scale

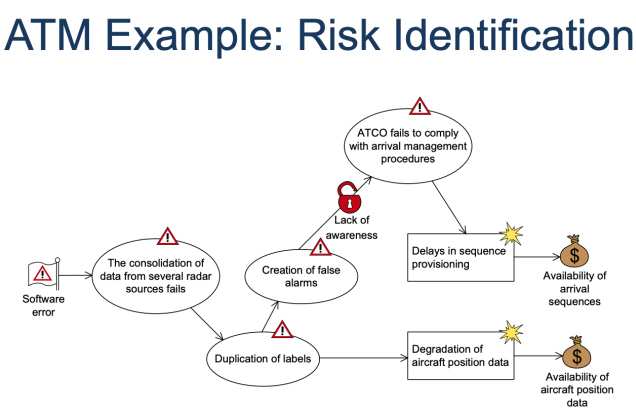
10.1.5 Step 5: Risk Identification using Threat Diagrams

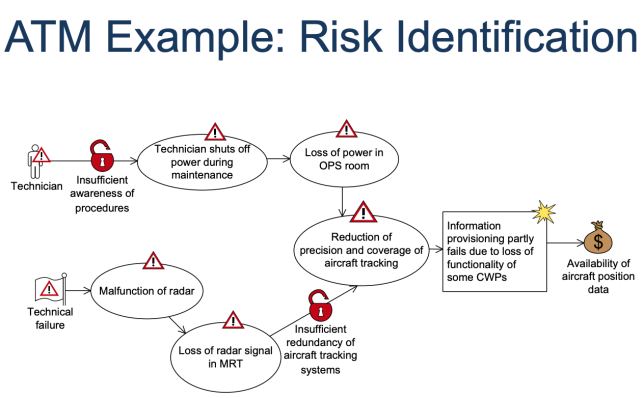
Objective

- Identify risk: where, when, why and how they may occur

Workshop conducted as a brainstorming session

- Involving people of different background
- Assets and high-level analysis as starting point
- Threats, threat scenarios, vulnerabilities and unwanted incidents documented on-the-fly using threat diagrams





10.1.6 Step 6: Risk Estimation Using Threat Diagrams

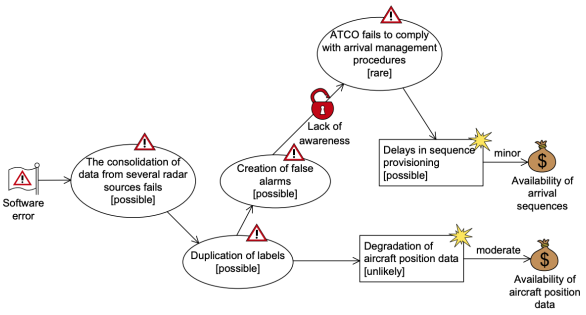
Objective

- Determine the level of identified risks

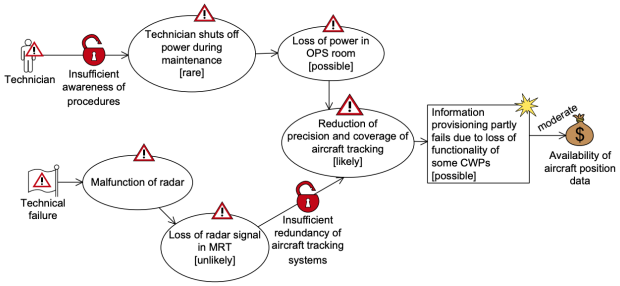
Workshop

- Involving people of different background
- Walk-through of threat diagrams
- Likelihood estimates on threat scenarios, unwanted incidents and relations between them
- Consequence estimates on relation between unwtanted incidents and assets

ATM Example: Risk Estimation



ATM Example: Risk Estimation



10.1.7 Step 7: Risk Evaluation Using Risk Diagrams

Objective

- Determine which risks are unacceptable and must be evaluated for treatment

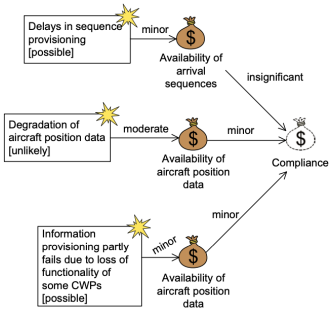
Off-line activity

- Calculate risk levels from estimates
- Present risks in risk diagrams

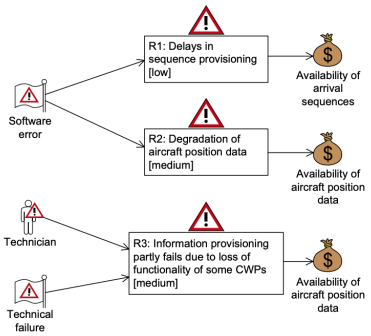
Assess potential impact of identified risk

- Risks that accumulate
- Risks with respect to indirect assets

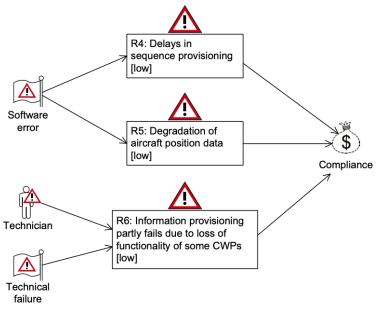
ATM Example: Indirect Assets



ATM Example: Risk Diagrams



ATM Example: Risk Diagrams



ATM Example: Risk Evaluation

Likelihood \ Consequence	Insignificant	Minor	Moderate	Major	Catastrophic
Rare					
Unlikely		R5	R2		
Possible	R4	R1, R6	R3		
Likely					
Certain					

- Risk levels are calculated using the risk matrix
- The risk matrix moreover serves as the risk evaluation criteria

10.1.8 Step 8: Risk Treatment Using Treatment Diagrams

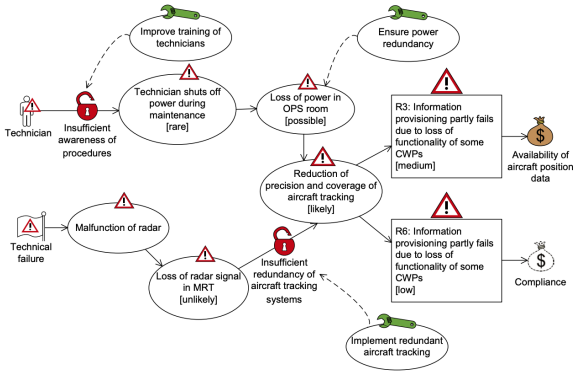
Objective

- Identify cost effective treatments for unacceptable risks

Workshop with brainstorming session

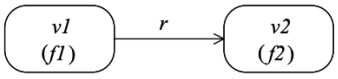
- Involving people of different background
- Walk-through of threat diagrams
- Identify treatments to unacceptable risks

ATM Example: Treatment Diagram



10.2 Frequency calculation

CORAS relation



vertex v1 is either a threat scenario or an unwanted incident
vertex v2 is either a threat scenario or an unwanted incident
f1, f2 are frequencies
r is a conditional ratio

Given *f1* and *r*, what do we know about *f2*?

Frequency of vertex

$$v(f)$$

the vertex v occurs with frequency f

Conditional ratio of relation

$$v \xrightarrow{r} v'$$

an occurrence of vertex v will lead to an occurrence of vertex v' with conditional ratio r

Occurrences due to

$$v_1 \sqcap\!| \, v_2$$

the vertex representing occurrences of vertex v2 that are due to an occurrence of vertex v1

Aggregation

$$v_1 \sqcup v_2$$

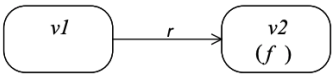
the vertex representing an occurrence of vertex v1 or an occurrence of vertex v2

Leads-to rule

If v1 occurs with frequence f and v1 leads-to v2 with consitional ratio r, then the number of occurrences of v2 due to v1 is f multiplied by r

$$\frac{H \vdash v_1(f) \quad H \vdash v_1 \xrightarrow{r} v_2}{H \vdash v_1 \sqcap\!| \, v_2(f \cdot r)}$$

CORAS initiate relation



vertex v1 is a threat
vertex v2 is either a threat scenario or an unwanted incident
r, f are frequencies

Given *r*, what do we know about *f* ?

Initiate rule

If v1 initiates v2 with frequency r, then the number of occurrences of v2 due to v1 is r

Special case of the leads-to rule

$$\frac{H \vdash v_1 \xrightarrow{r} v_2}{H \vdash v_1 \sqcap\!| \, v_2(\quad r)}$$

Aggregation rule

If
- v1 occurs with frequence f1
- v2 occurs with frequence f2
- an occurrence of v1 cannot be an occurrence of v2
- an occurrence of v2 cannot be an occurrence of v1

then
- v1 or v2 occurs with frequency f1+f2

$$\frac{H \vdash v_1(f_1) \quad H \vdash v_2(f_2) \quad s(v_1) \cap s(v_2) = \varnothing}{H \vdash v_1 \sqcup v_2(f_1 + f_2)}$$

11 Lecture 10: Security Risk Assessment III

Overview

- Consequence calculation
- Three perspectives on change
- Risk graphs with change
- CORAS instantiation
- Practical example

11.1 Consequence calculation

11.1.1 Pre-requisite

Not possible unless the relevant consequence scales have been concerted into a common scale. In the following we assume consequence is measured in terms of

Average loss inn EURO per occurrence

11.1.2 Rule of aggregation of consequence

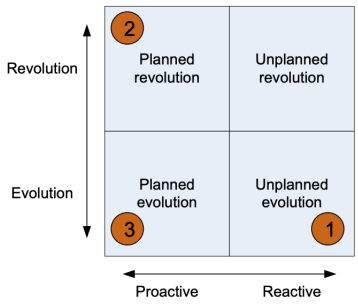
IF

- incident v1 occurs with frequency f1 and consequence c1
- incident v2 occurs with frequency f2 and consequence c2
- incident v1 and incident v2 are separate

THEN

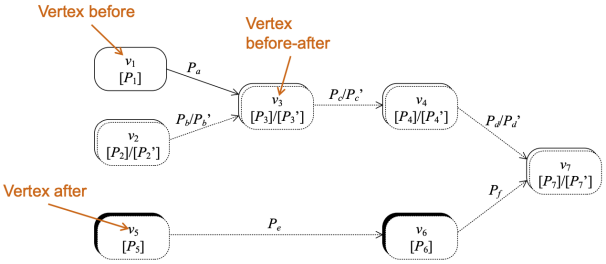
- the aggregated incident occurs with consequence (f1*c1+f2*c2)/(f1+f2)

11.2 Three perspective changes

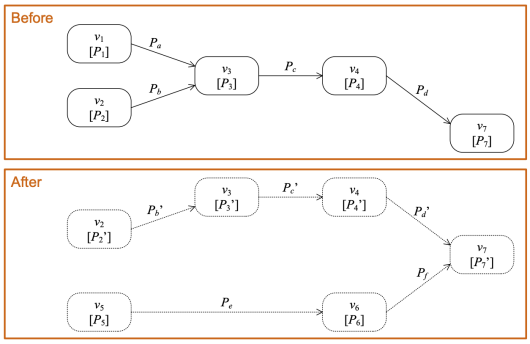


1. The maintenance (a posteriori) perspective
2. The before-after (a priori) perspective
3. The continuous evolution perspective

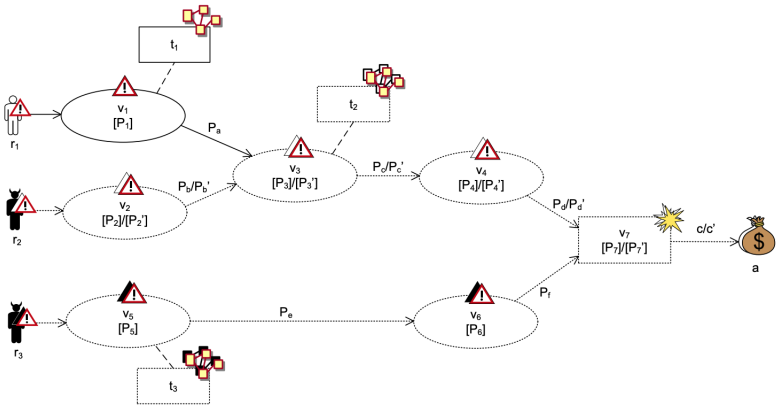
11.3 Risk Graphs with Change



Two views on risk graphs with change



11.4 CORAS Instantiation



11.5 Practical Example: ATM

11.5.1 Changes

Current characteristic of ATM

- Limited interaction with external world
  - Limited security problems in relation to information flow to and from the environment
- Humans at the centre
  - Limited role of automated decision support systems and tools

Change in European ATM

- Intoduction of new information systems and decision support systems
- Reorganization of services

11.5.2 Target of Analysis

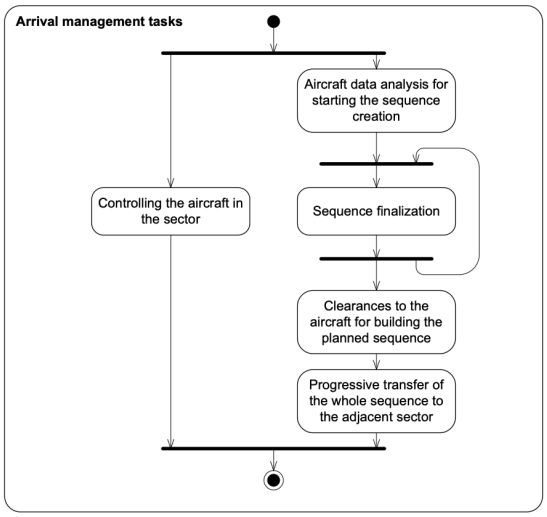
Arrival management and the role of air traffic controllers (ATCOs) in the area control centre (ACC)
The introduction of AMAN and ADS-B

- Arrival manager (AMAN) is a decision support tool for the automation of ATCO tasks in the arrival management
- Automatic Dependent Surveillance-Broadcast (ADS-B) is a cooperative GPS-based surveillance technique where aircrafts constantly broadcast their position to the ground and to other aircrafts

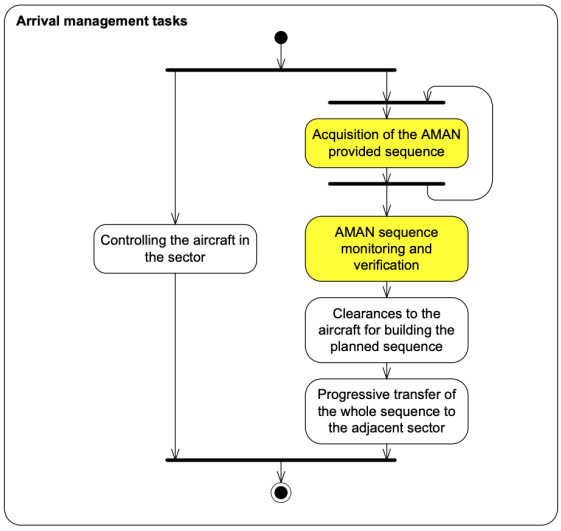
11.5.3 Focus of Analysis

- Before changes:
  - Information provision (availability)
  - Compliance with regulation
- Additional concerns after changes:
  - Information protection (confidentiality)

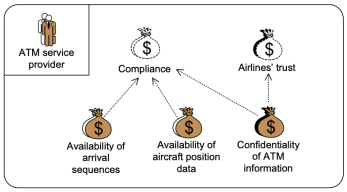
Target Before



Target After



Assets Before-After



- Party remains the same under change
- Direct asset Confidentiality of ATM information is considered only after changes
- Indirect asset Airlines' trust is considered only after changes

Consequence Scales

Confidentiality

Consequence	Description
Catastrophic	Loss of data that can be utilized in terror
Major	Data loss of legal implications
Moderate	Distortion of air company competition
Minor	Loss of aircraft information data
Insignificant	Loss of publically available data

Availability

Consequence	Description
Catastrophic	Catastrophic accident
Major	Abrupt maneuver required
Moderate	Recovery from large reduction in separation
Minor	Increasing workload of ATCOs or pilots
Insignificant	No hazardous effect on operations

Likelihood Scale

Likelihood	Description
Certain	A very high number of similar occurrences already on record; has occurred a very high number of times at the same location/time
Likely	A significant number of similar occurrences already on record; has occurred a significant number of times at the same location
Possible	Several similar occurrences on record; has occurred more than once at the same location
Unlikely	Only very few similar incidents on record when considering a large traffic volume or no records on a small traffic volume
Rare	Has never occurred yet throughout the total lifetime of the system

Risk Evaluation Criteria

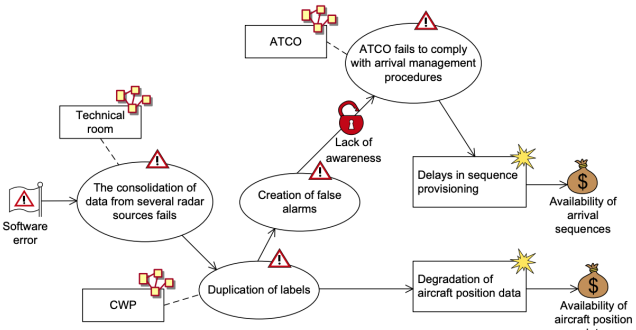
Likelihood \ Consequence	Insignificant	Minor	Moderate	Major	Catastrophic
Rare					
Unlikely					
Possible					
Likely					
Certain					

- **High risk:** Unacceptable and must be treated
- **Medium risk:** Must be evaluated for possible treatment
- **Low risk:** Must be monitored

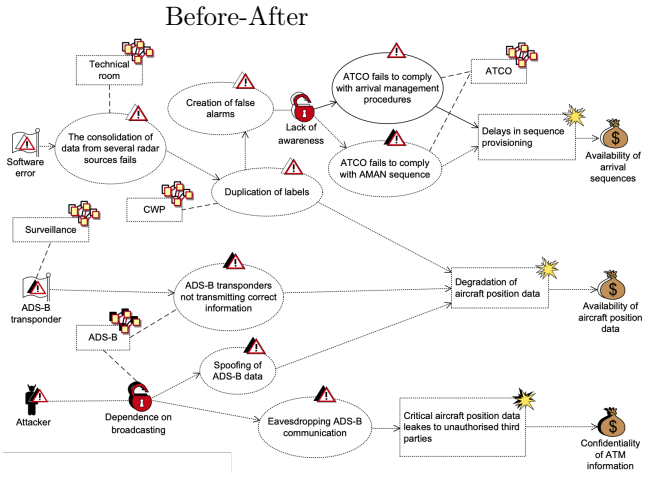
Note: Also the evaluation criteria may change

11.6 CORAS Step 5 - Risk Identification

Before

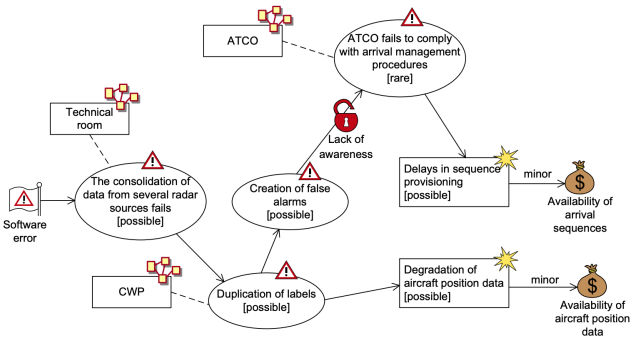


Before-After

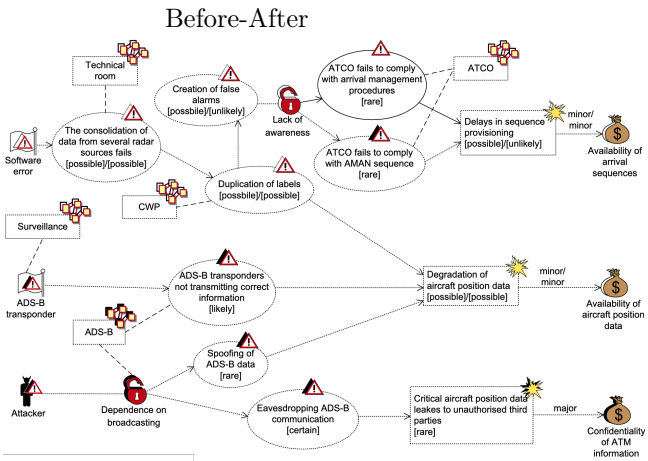


11.7 CORAS Step 6 - Risk Estimation

Before



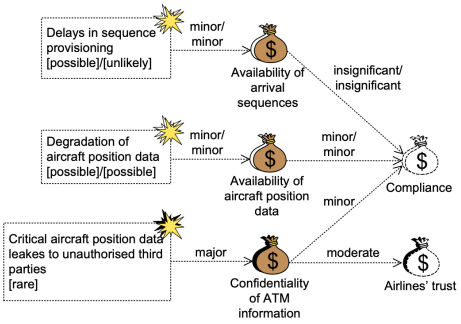
Before-After



11.8 CORAS Step 7 - Risk Evaluation

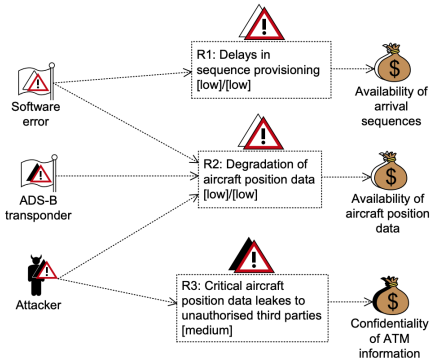
Indirect Assets

- Before-After



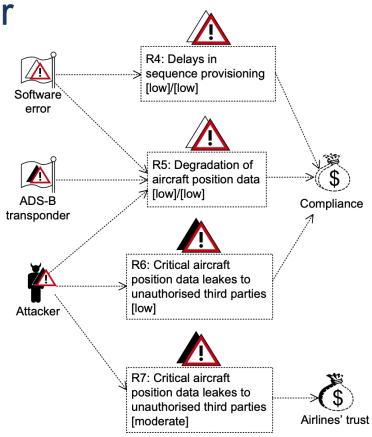
Risk Diagram

- Before-After



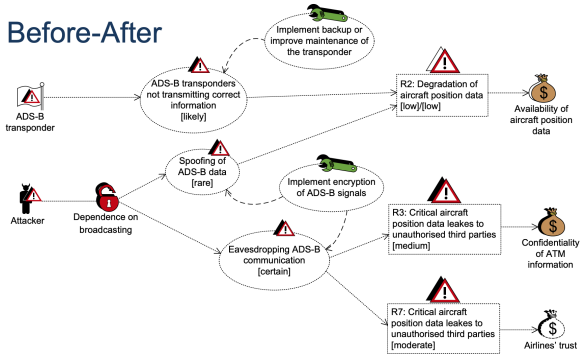
Risk Diagram

- Before-After



Treatment Diagram

- Before-After



Risk Evaluation

Consequence

Likelihood	Insignificant	Minor	Moderate	Major	Catastrophic
Rare		**R6**	**R7**	**R3**	
Unlikely	**R4**	**R1**			
Possible	*R4*	*R1*, ***R2***, ***R5***			
Likely					
Certain					

- Legend:
  - *Italic* denotes risk before
  - **Bold** denotes risk after

12 Lecture 11: Uncertainty, Subjectivity, Trust and Risk How It All Fits Together

Overview

- Uncertainty
- Subjectivity versus Objectivity
- Risk Management
- Trust Management
- Trust versus Risk
- The Three Focal Points of Trust Management

12.1 Uncertainty

Epistemic uncertainty: Uncertainty due to ignorance and lack og evidence.

Aleatory uncertainty: Uncertainty due to inherent randomness of systems.

We try to reduce the epistemic uncertainty with risk assesment.

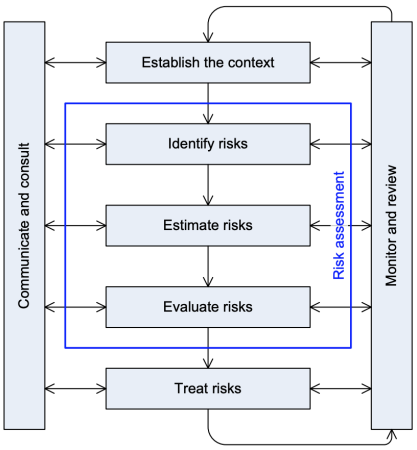
12.2 Subjectivity versus Objectivity

Subjective: pertains to the subject and how the subject perceives an object; depends on the subject's perception associated with the false and the possibility of wrong perceptions.

Objective: pertains to the existence of an object outside the consciousness and independent of the subject's perception of the object; associated with the true and the factual reality.

12.3 Risk Management

risk management = coordinated activities to direct and control an organization with regard to **risk** ISO 31000



12.3.1 Definition of risk from ISO 31000

Risk = effect of uncertainty of objectives

- NOTE 1 An effect is a deviation from the expected — positive and/or negative
- NOTE 2 Objectives can have different aspects (such as financial, health and safety, and environmental goals) and can apply at different levels (such as strategic, organization-wide, project, product and process)
- NOTE 3 Risk is often characterized by reference to potential events and consequences, or a combination of these
- NOTE 4 Risk is often expressed in terms of a combination of the consequences of an event (including changes in circumstances) and the associated likelihood of occurrence
- NOTE 5 Uncertainty is the state, even partial, of deficiency of information related to, understanding or knowledge of an event, its consequence, or likelihood

12.3.2 Excerise I

How would you represent risk in a sequence diagram?

12.4 Trust management

Trust management is a label for a diversity of approaches.

Shared ground: Strong relation between *trust* on the one hand and *risk* on the other.

Often unclear how, exactly trust relates to risk

12.4.1 Trust

Trust is a relationship between to entities

- Trustor (the trusting party)
- Trustee (the trusted party)

Trust (or, symmetrically, distrust) is the subjective probability with which the trustor expects that the trustee performs a given action on which its welfare depends

12.5 Trust versus Risk

In case the trustee performs as expected it may have a positive effect on the welfare of the trustor, otherwise it may have a negative effect

The positive and negative outcomes corresponds to opportunity and risk

There is always a possibility of deception or betrayal, which means that there is an inevitable relation between trust and risk

Trust is always related to opportunity; the trustor may be willing to accept the risk considering the involved opportunities

12.5.1 Trust aspects

Trust is *subjective*

- Trust is a belief

Trust is a *probability* ranging from 0 to 1

- 1 denotes complete trust
- 0 denotes complete distrust

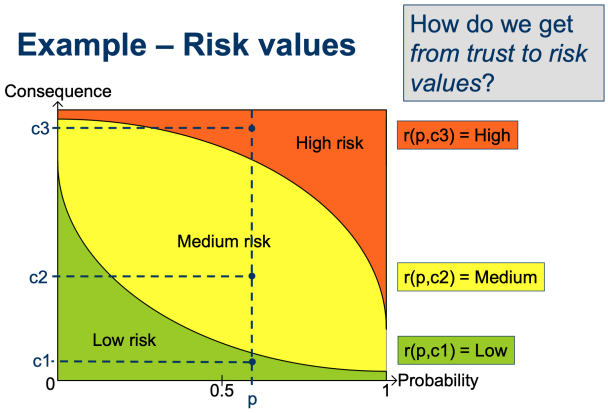
- t is the *trust threshold*
- d is the *distrust threshold*

12.5.2 Risk

Risk value is given as a function from both

- *Probability*: Probability of incident to occur
- *Consequence*: Harm of incident on asset

r: P * C → RV



12.5.3 Exercise II

How would you define the relationship between trust and risk?

12.5.4 Trust is not inverse proportional to risk

It is often suggested that trust is inverse to risk, i.e.

- High trust means low risk
- Low trust (i.e. distrust) means high risk

But then *consequence* is not considered

- High trust may imply high values at stake, *increasing* the risk
- Distrust may imply no or low values at stake, *minimizing* the risk

High trust only means low probability of a harmful incident

12.5.5 Trust is not proportional to risk

Some models suggest that trust is risk acceptance

- "You are prepared to accept risks if you trust the entities that can expose them"
- This means that high trust implies high risk

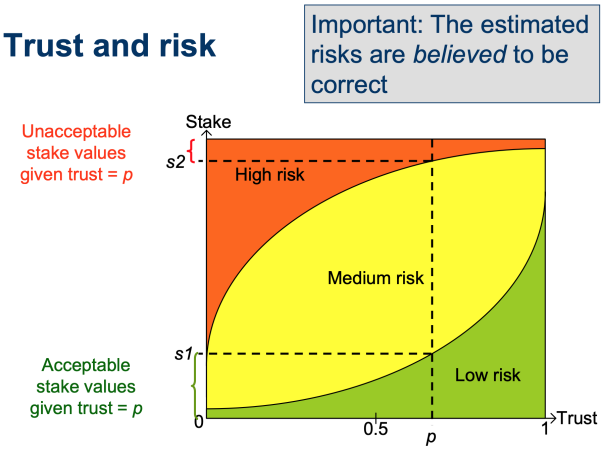
Trust is then wrongly identified with the stake value
The *probability* is not really considered

High trust only means that we may accept high stake

12.5.6 Estimating risk from trust value

- Risk value is derived functionality from probability and consequence; **r : P * C → RV**
- A subjective probability estimate **p** in a trust based transaction is not enough for estimating the risk
- We also need the consequence **c**, i.e. the value at stake
- The risk value for trust **p** is then **r ((1-p),c)**

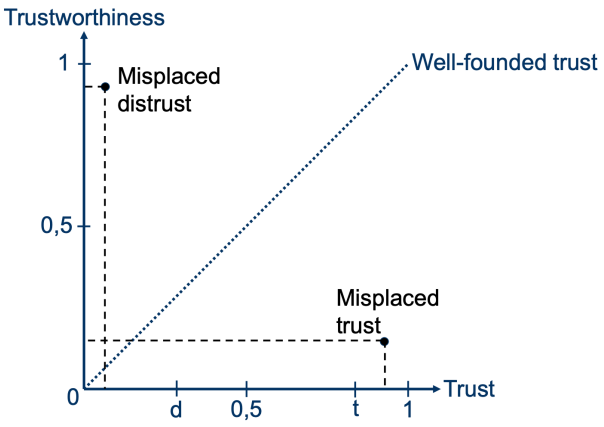
12.5.7 Well-founded Trust



12.5.8 Trustworthiness

- *Trust* (or, symmetrically, distrust) is the *subjective* probability by which the trustor expects that the trustee performs a given action on which its welfare depends
- *Trustworthiness* is the *objective* (or factual) probability by which the trustee performs a given action on which the welfare of the trustor depends
- Well-founded trust: The trustor knows the trustworthiness of the trustee, i.e. trust = trustworthiness

12.5.9 Well-founded Trust



12.6 Three focal points in trust management

- Trust management on behalf of the trustor
- Trust management on behalf of the trustee
- Trust management on behalf of system owner

12.6.1 Focal points illustrated

Factual reality	**Trust** Subjective probability of Bob being just and deck being fair	**Trustworthiness** Objective probability of Bob being just and deck being fair
Uncovering factual reality	**Target of investigation** Alice	**Target of investigation** Bob and deck