CS-3004
SOFTWARE DESIGN AND ANALYSIS

RUBAB JAFFAR

RUBAB.JAFFAR@NU.EDU.PK

Interaction diagrams

TODAY'S OUTLINE

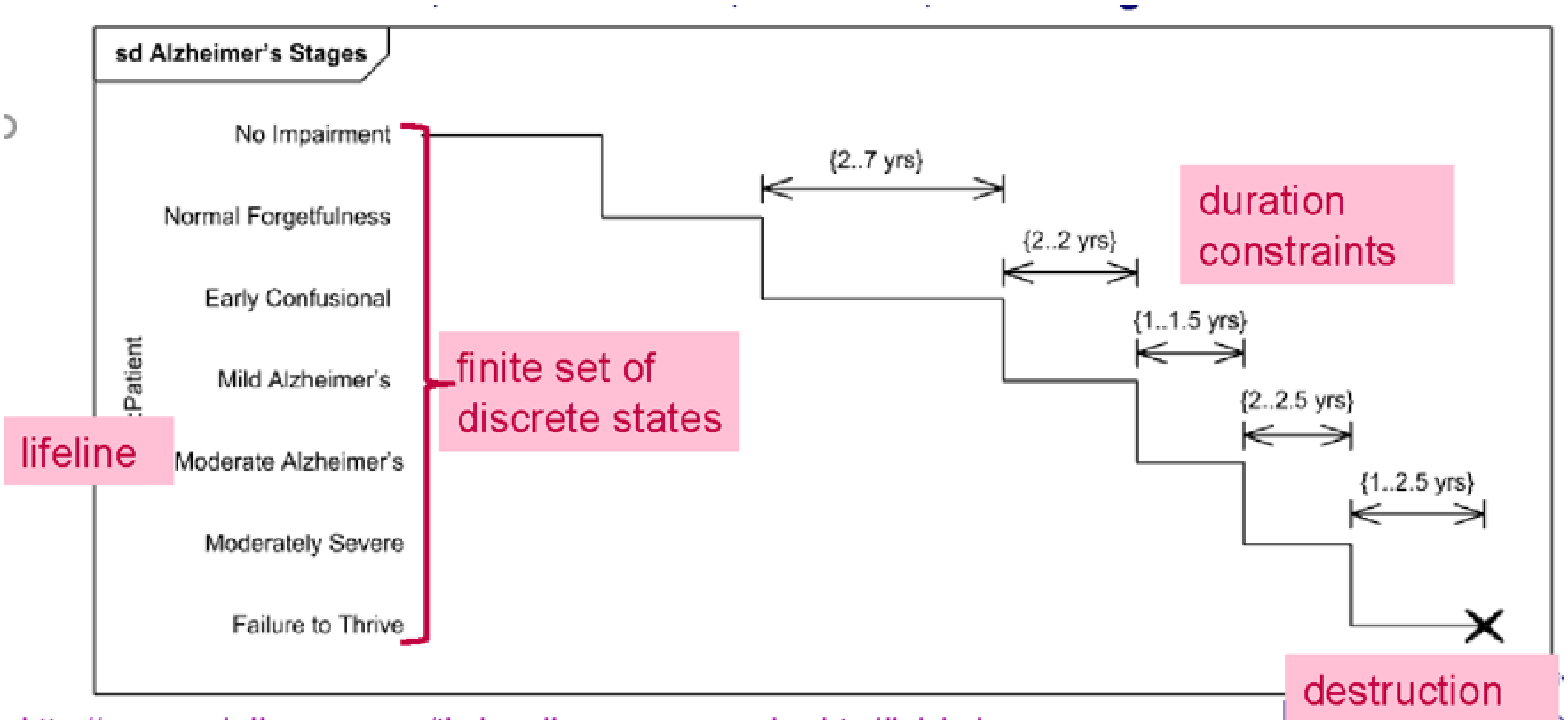
- Interaction Diagram
  - Timing Diagram
- Elements of Timing Diagram
- Timing Diagram Notations
- Examples
- 4+1 view model
- Homogenization of classes

TIMING DIAGRAMS

- A timing diagram allows you to show the interaction of objects and changes in state for those objects along a time axis.
- A timing diagram provides a convenient way to show active objects and their state changes during their interactions with other active objects and system resources.
- The X-axis of the timing diagram has the time units, while the Y-axis shows the objects and their states.
- Timing diagrams describe behavior of both individual **classifiers** & **interactions of classifiers**, focusing attention on time of events causing changes in the modeled conditions of the lifelines.

TIMING DIAGRAM ELEMENTS

- Timing Diagrams are Interaction diagram for reasoning about time.
- **Basic elements**: lifelines, states, duration/time constraints, destruction, events, messages



lifeline

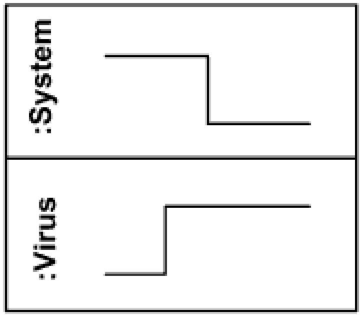
finite set of discrete states

duration constraints

destruction

LIFELINE

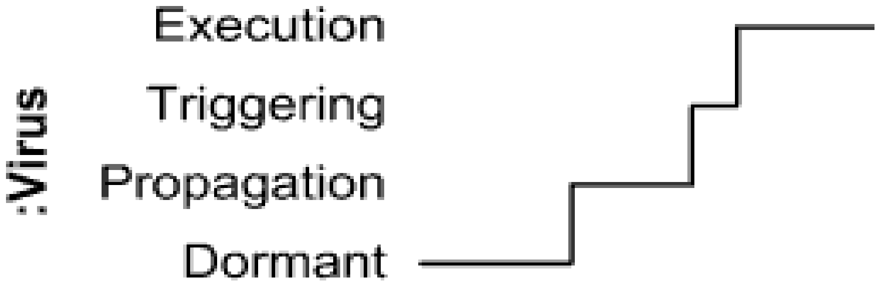
- **Lifeline** is a **named element** which represents an **individual participant** in the interaction. Lifelines represent **only one** interacting entity.
- Lifeline on the timing diagrams is represented by the **name** of classifier or the instance it represents. It could be placed inside diagram frame or a "swimlane".



representing instances of System and Virus

STATE OR CONDITION TIMELINE

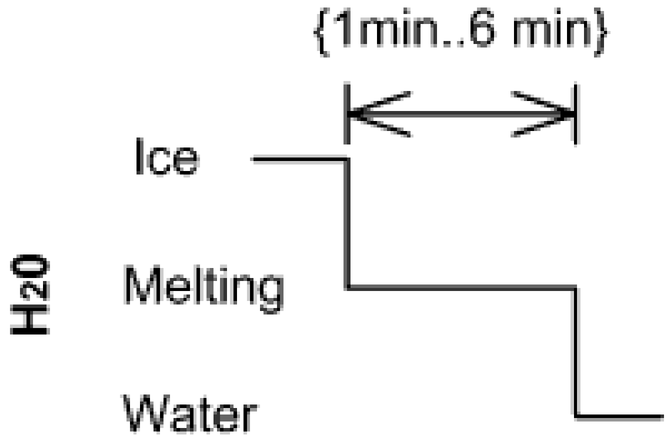
- Timing diagram could show **states** of the participating **classifier** or attribute, or some testable **conditions**, such as a discrete or enumerable value of an attribute.



- *Timeline shows Virus changing its state between Dormant, Propagation, Triggering and Execution state*

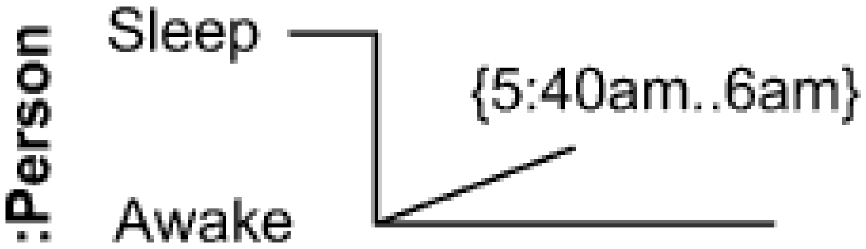
DURATION CONSTRAINT

- **Duration constraint** is an **interval constraint** that refers to a **duration interval**. The duration interval is duration used to determine whether the constraint is satisfied.
- E.g., *Ice should melt into water in 1 to 6 minutes*



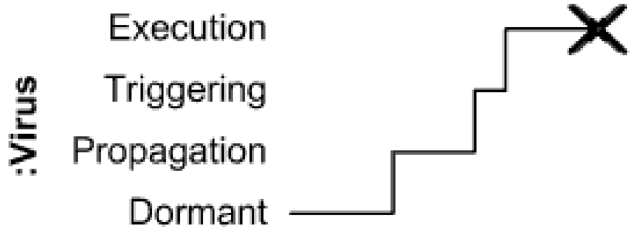
TIME CONSTRAINT

- **Time constraint** is an **interval constraint** that refers to a **time interval**. The time interval is time expression used to determine whether the constraint is satisfied.
- Typically this graphical association is a small line, e.g., between an occurrence specification and a time interval.
- E.g., *Person should wake up between 5:40 am and 6 am*

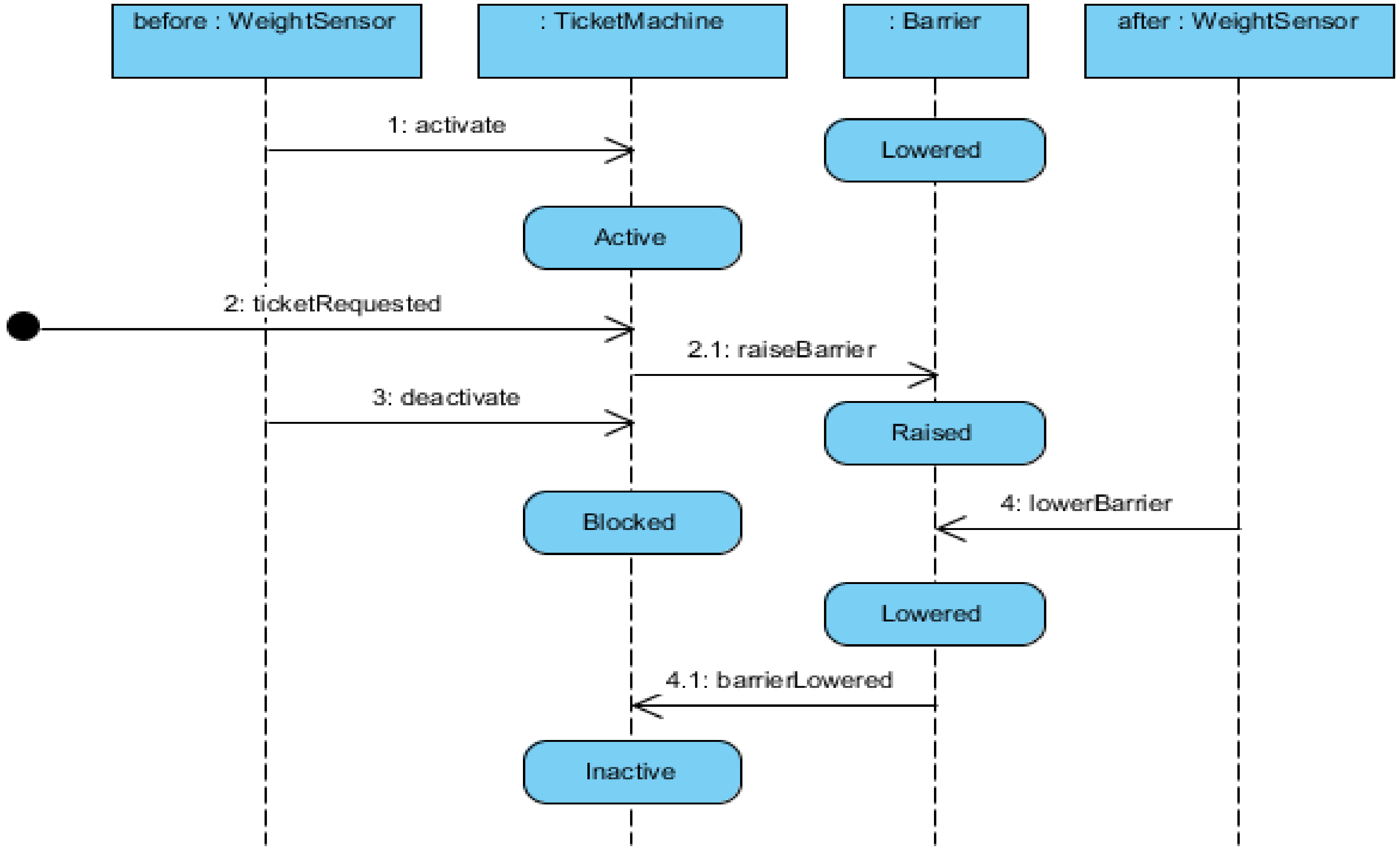


DESTRUCTION OCCURRENCE

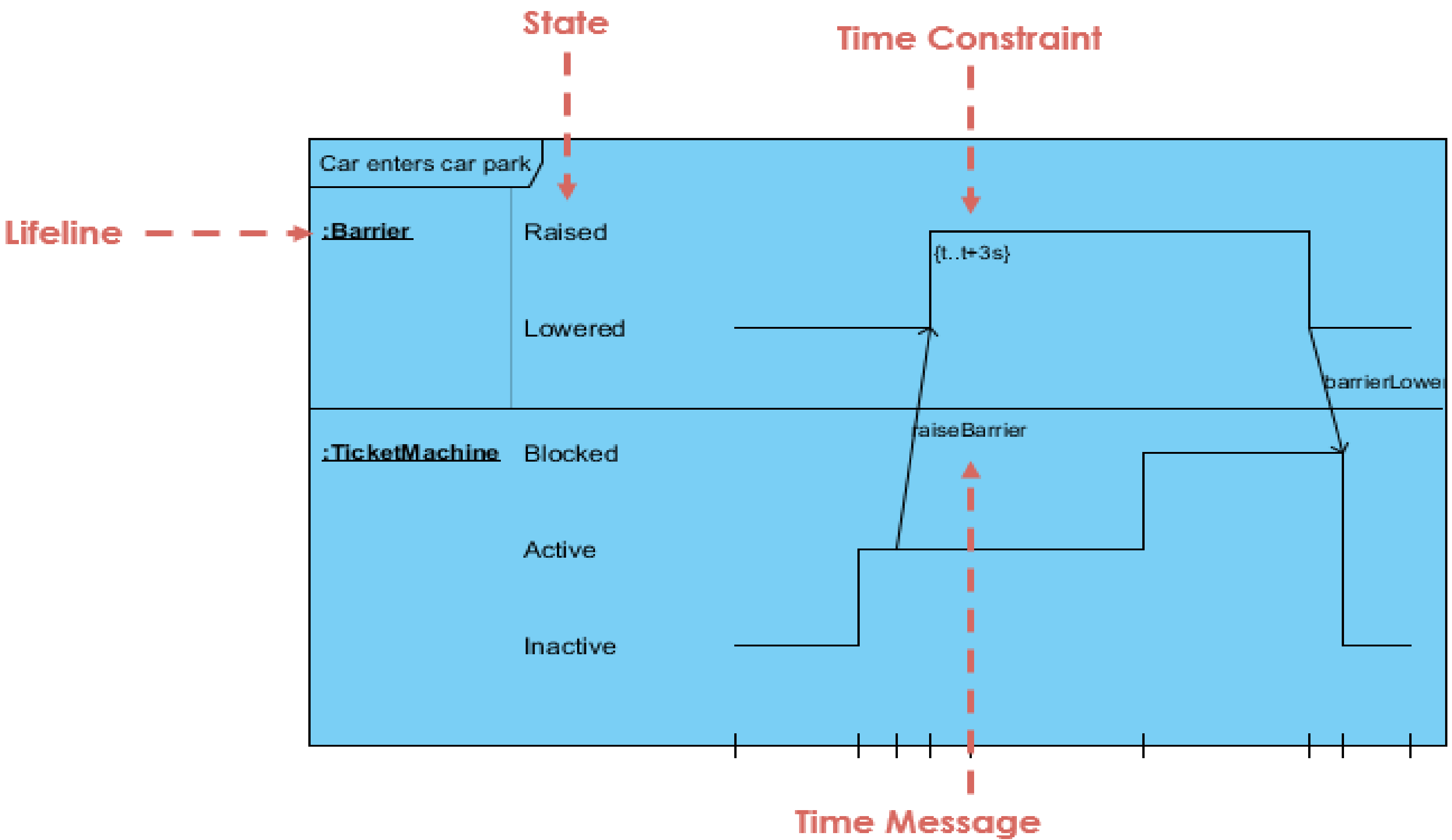
- **Destruction occurrence** is a **message occurrence** which represents the destruction of the instance described by the **lifeline**.
- It may result in the subsequent destruction of other objects that this object owns by **composition**.
- No other occurrence may appear after the destruction event on a given lifeline.
- ***Notation***
- The destruction event is depicted by a cross in the form of an **X** at the end of a timeline.



CARPARK EXAMPLE

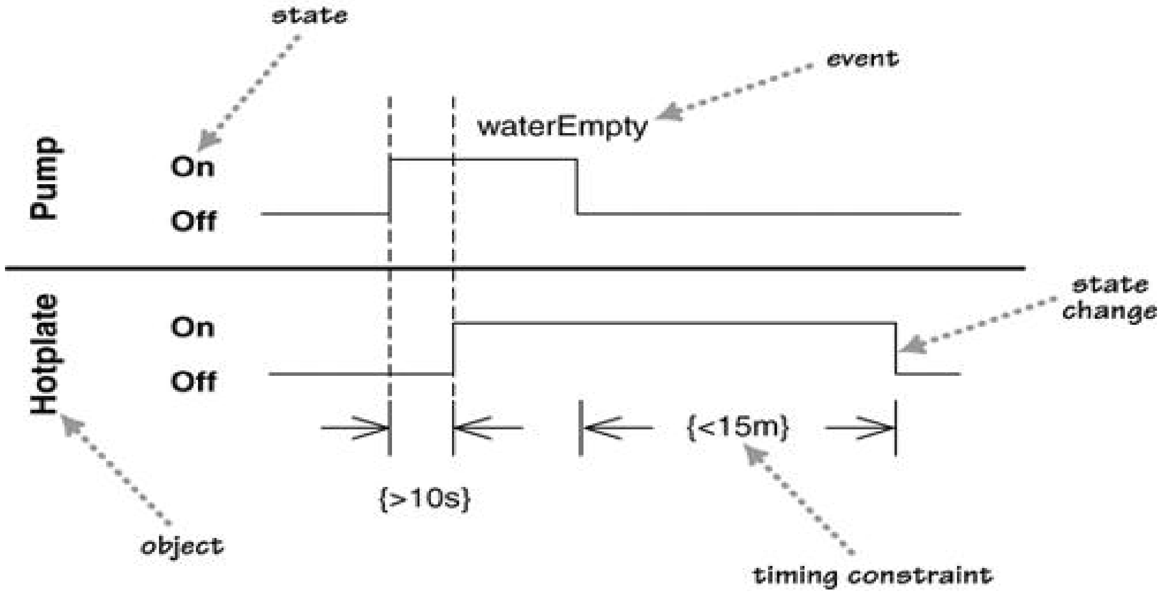


CARPARK EXAMPLE



EX 1: COFFEE POT

- Let's take a simple scenario based on the pump and hotplate for a coffee pot. Let's imagine a rule that says that at least 10 seconds must pass between the pump coming on and the hotplate coming on. When the water reservoir becomes empty, the pump switches off, and the hotplate cannot stay on for more than 15 minutes more.

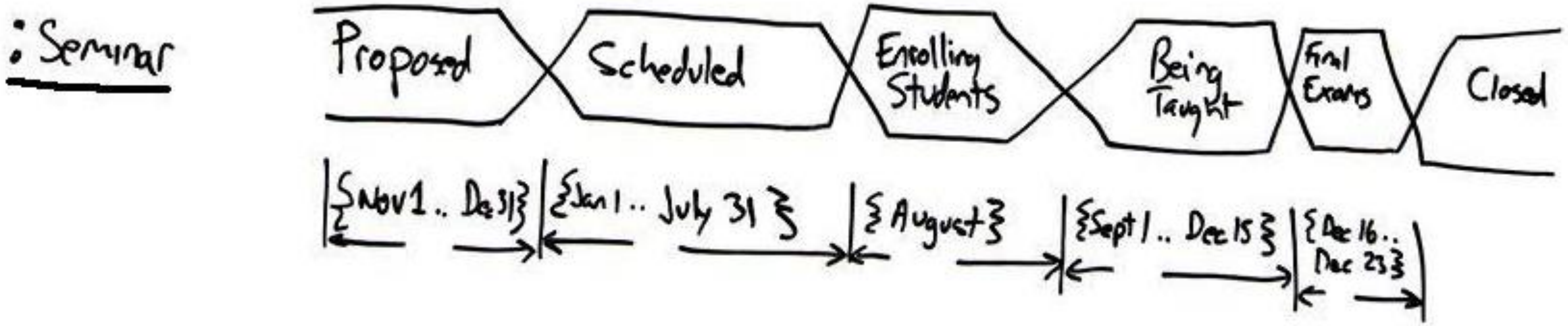


TIMING DIAGRAM NOTATIONS

- There are two basic flavors of timing diagram,
  - Concise notation
  - Robust notation
- Showing states as lines – They are known as State Lifeline diagram or robust notations
- Showing states as areas - They are known as Value Lifeline diagram or consise notations

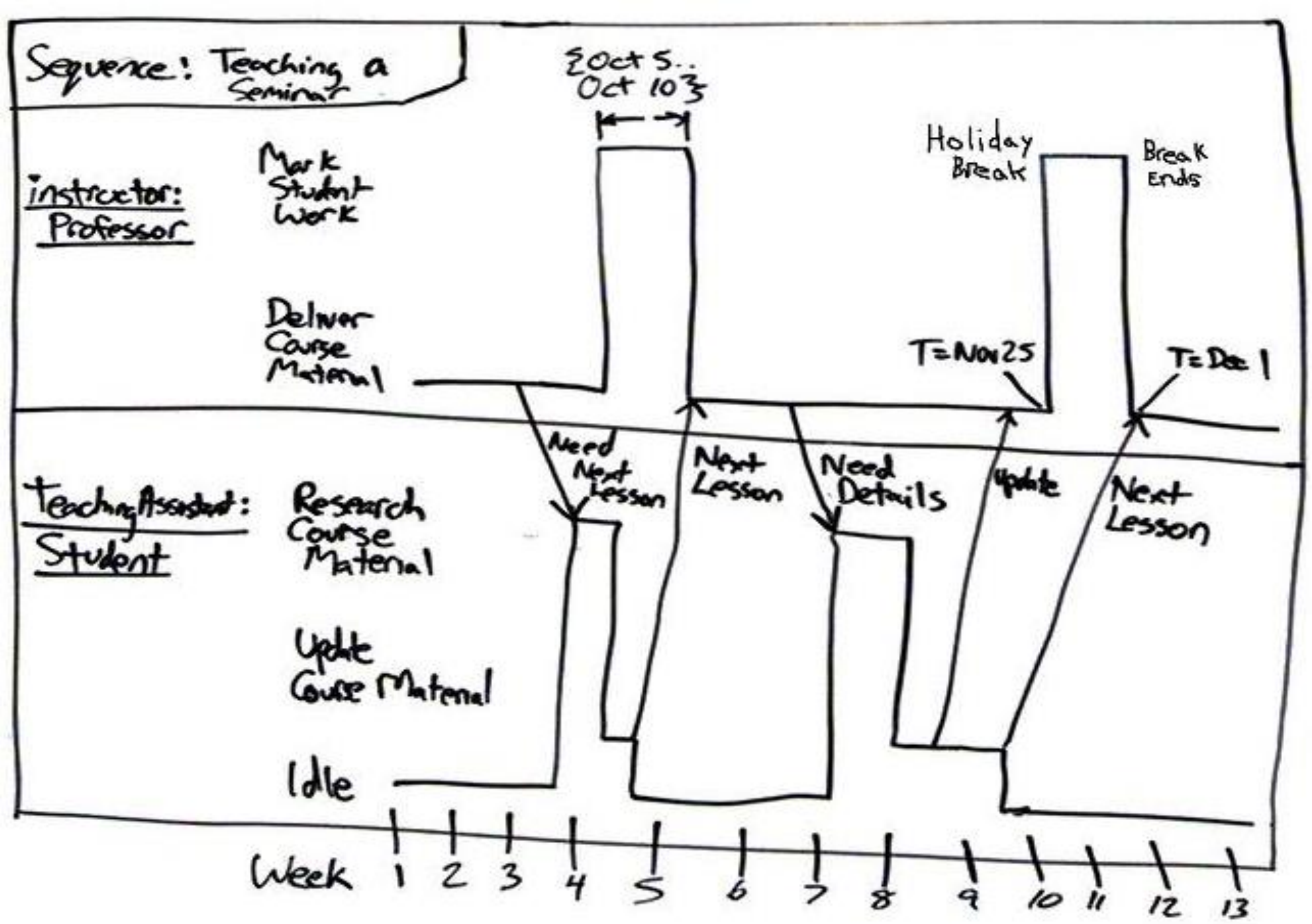
TIMING DIAGRAM-CONCISE NOTATION

- Life Cycle of Seminar
- The critical states that the seminar exhibits
  - *Proposed,*
  - *Scheduled,*
  - *Enrolling Students,*
  - *Being Taught,*
  - *Final Exams,*
  - *Closed*

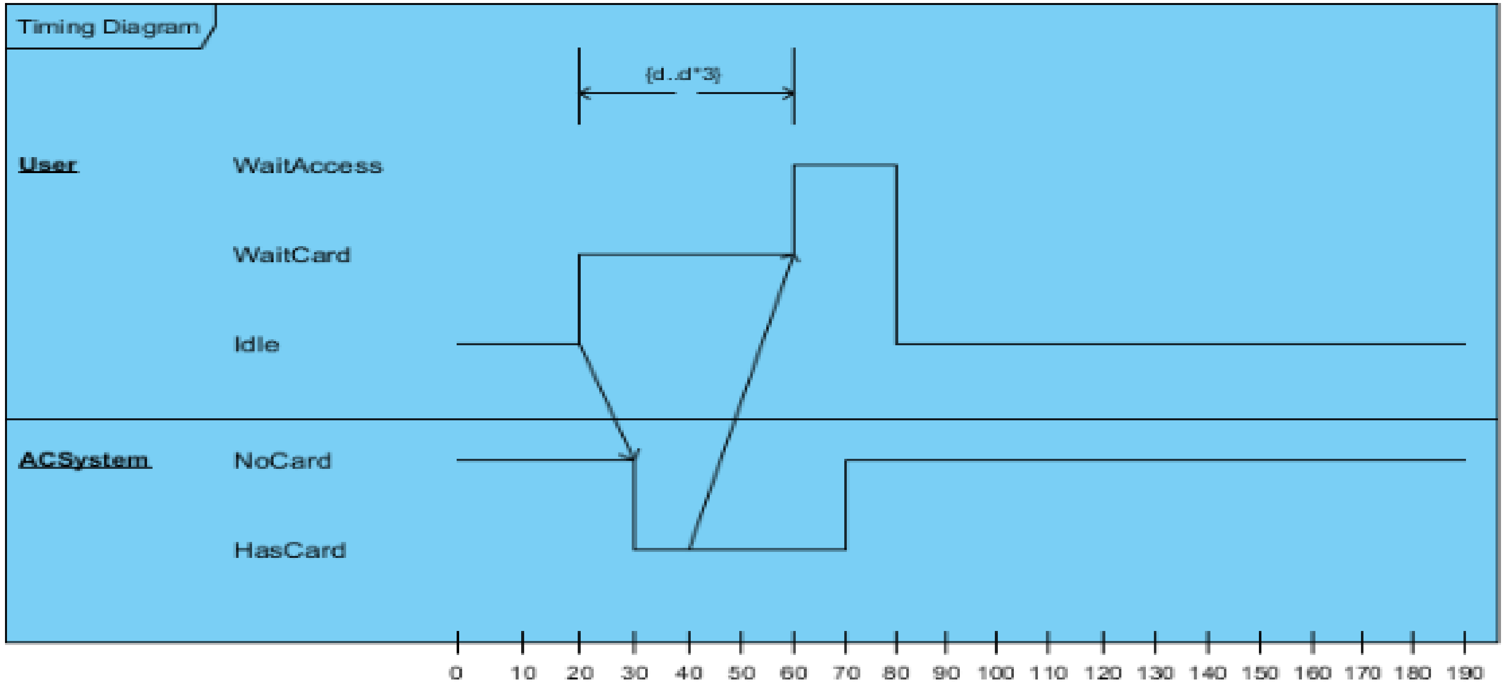


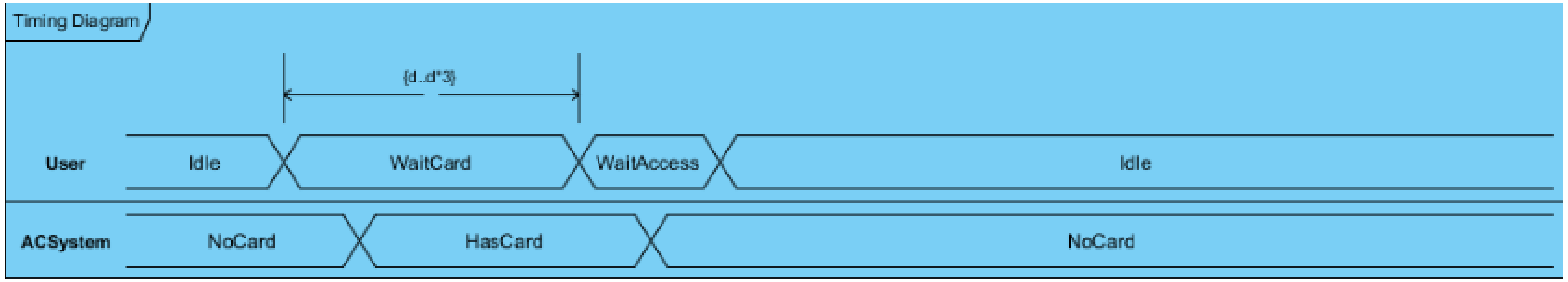
TIMING DIAGRAM-ROBUST NOTATION

- **Happenings while a seminar is being taught.**
- The professor delivers the seminar and marks student work
- The teaching assistant develops the course material by doing the research on course material and then update the material just in time for it to be taught.
- Total duration of seminar is 13 weeks. Events/stimuli, such as *Holiday Break* and *Break Ends* indicate the reason for the change.
- The arrows between timelines are messages between the objects.


Sequence: Teaching a Seminar
{Oct 5.. Oct 10}
Holiday Break
Break Ends
instructor: Professor
Mark Student Work
Deliver Course Material
T=Nov25
T=Dec 1
Need Next Lesson
Next Lesson
Need Details
Update
Next Lesson
TeachingAssistant: Student
Research Course Material
Update Course Material
Idle
Week
1
2
3
4
5
6
7
8
9
10
11
12
13

ANOTHER EXAMPLE

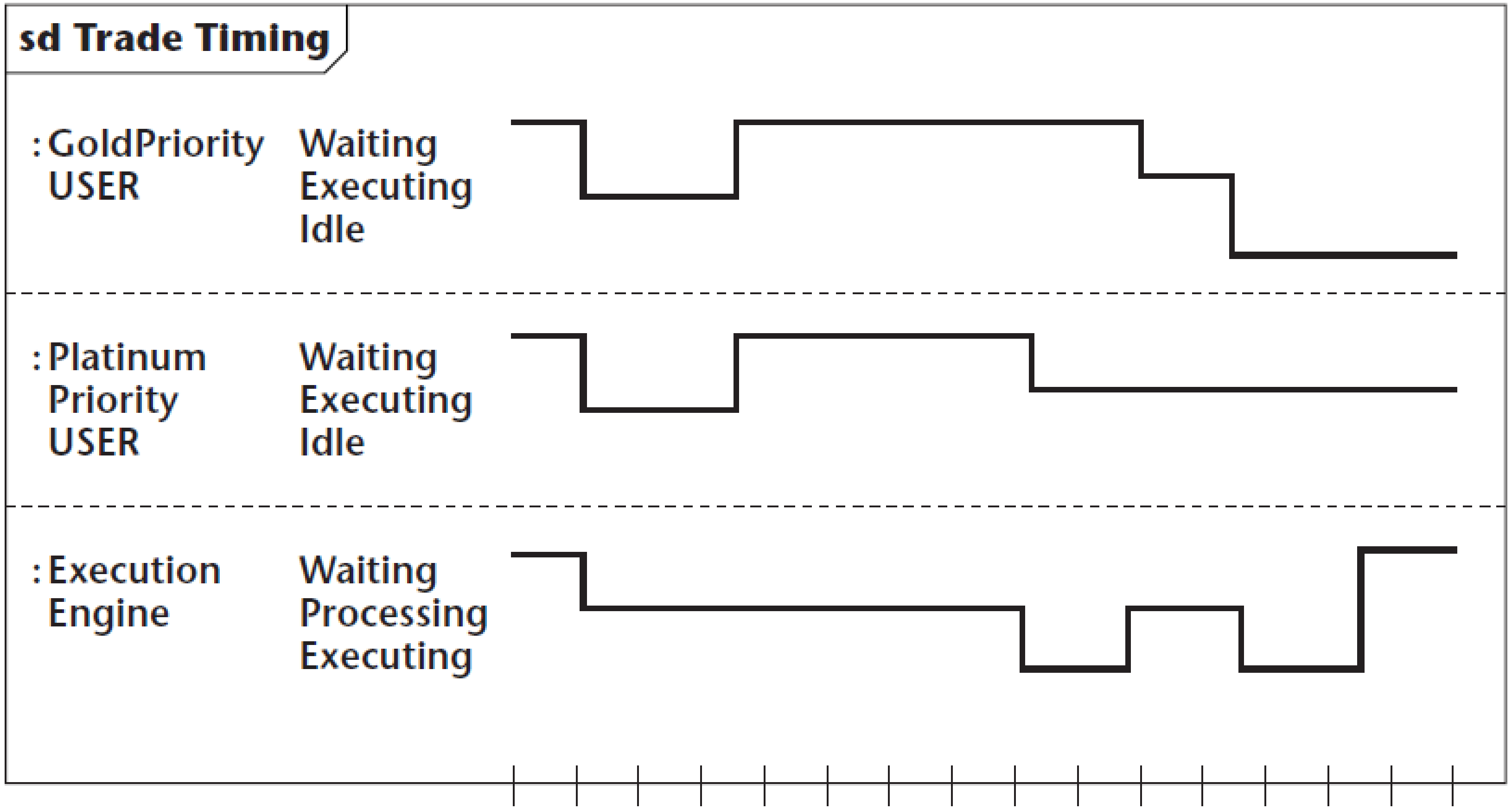




EXAMPLE 1

- Figure 6.19 shows a case where two active objects share a common resource. In this case, both objects show a trade that requires the execution engine for some time. One object is for a "platinum" user who is guaranteed a trade within 10 time units, and the other is a "gold" user who has no performance guarantee. The user objects have a waiting and executing state along with an idle state. The execution engine has a processing state that takes five time units for each trade and an executing state that takes two time units.

EXAMPLE 1

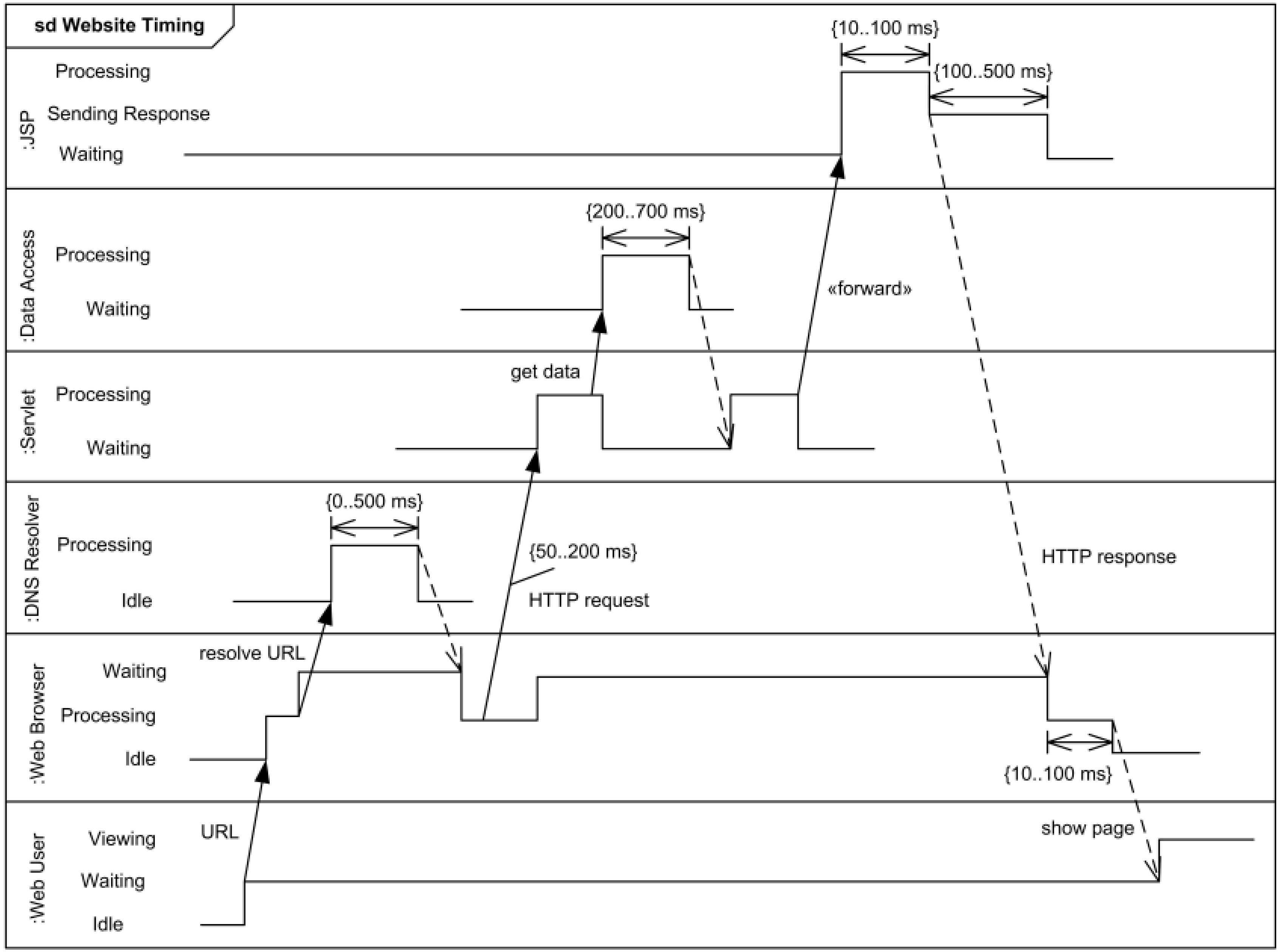

sd Trade Timing
:GoldPriority USER
Waiting
Executing
Idle
:Platinum Priority USER
Waiting
Executing
Idle
:Execution Engine
Waiting
Processing
Executing

EX 2: WEBSITE LATENCY

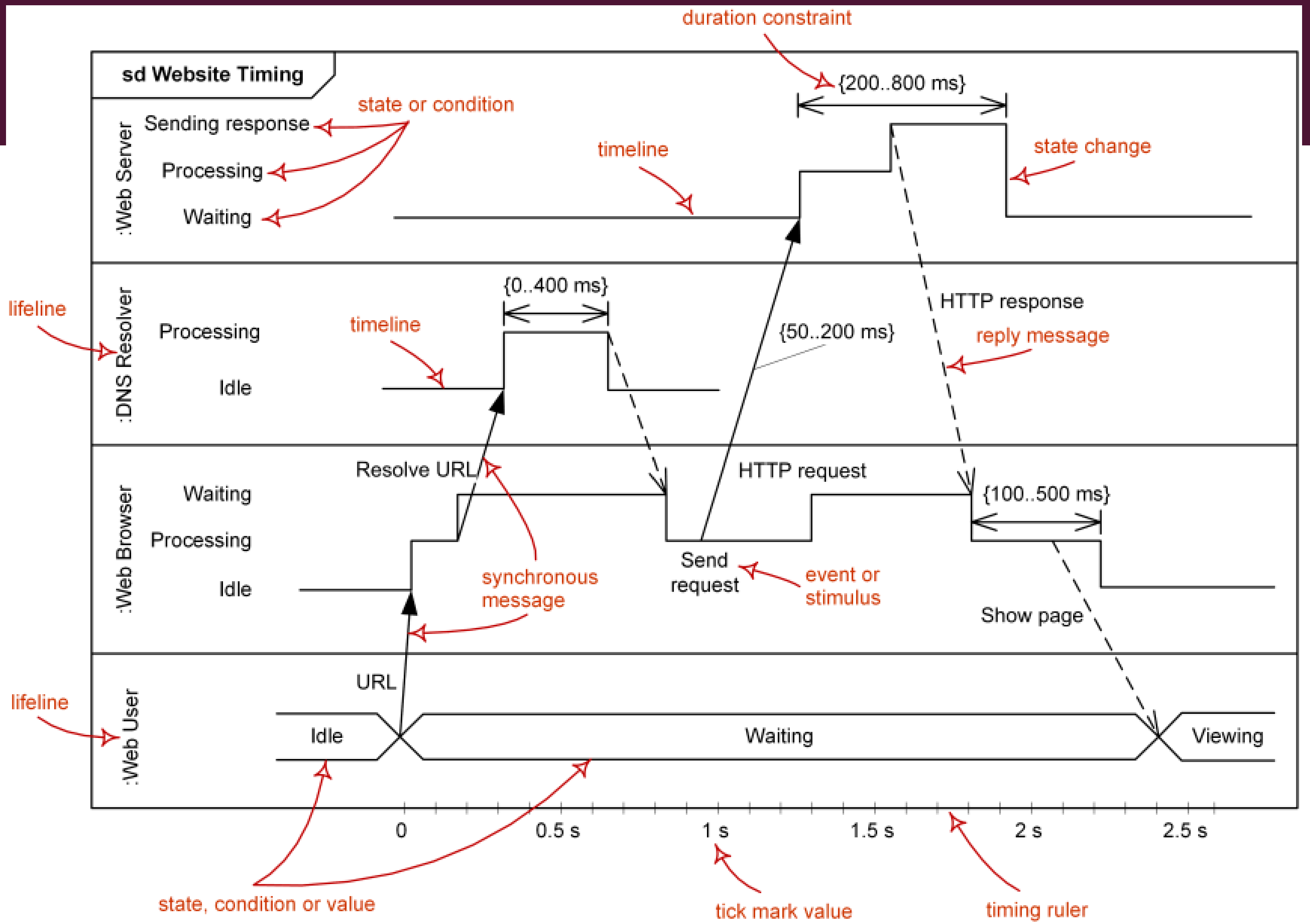
- An example of **timing diagram** which shows some **duration constraints** for a fabricated website to evaluate how long web user should wait to see something rendered on his/her display.
- After web user enters web page URL, the URL should be resolved to some IP address. DNS resolution can add some tangible waiting time to the response latency as perceived by user. Latency delays related to DNS resolution could range from 1 ms (local DNS cache) to several seconds.

EX 2: WEBSITE LATENCY

- With simple Model–View–Control (MVC) implementation, Java servlet gets control and requests some data from "model". Communication with data sources usually takes some discernible time. After data is received and processed, servlet forwards request processing to JSP ("view"). Buffered HTTP response is sent back to the browser.
- Web browser takes some time to process HTTP response and HTML page to start rendering the page view to the web client. (Note, that after that it could take even more time for the web browser to request other resources like CSS, JavaScript, images, which is not shown on the diagram.)


sd Website Timing
:JSP
Processing
Sending Response
Waiting
{10..100 ms}
{100..500 ms}
:Data Access
Processing
Waiting
{200..700 ms}
«forward»
:Servlet
Processing
Waiting
get data
:DNS Resolver
Processing
Idle
{0..500 ms}
{50..200 ms}
HTTP request
HTTP response
:Web Browser
Waiting
Processing
Idle
resolve URL
{10..100 ms}
:Web User
Viewing
Waiting
Idle
URL
show page

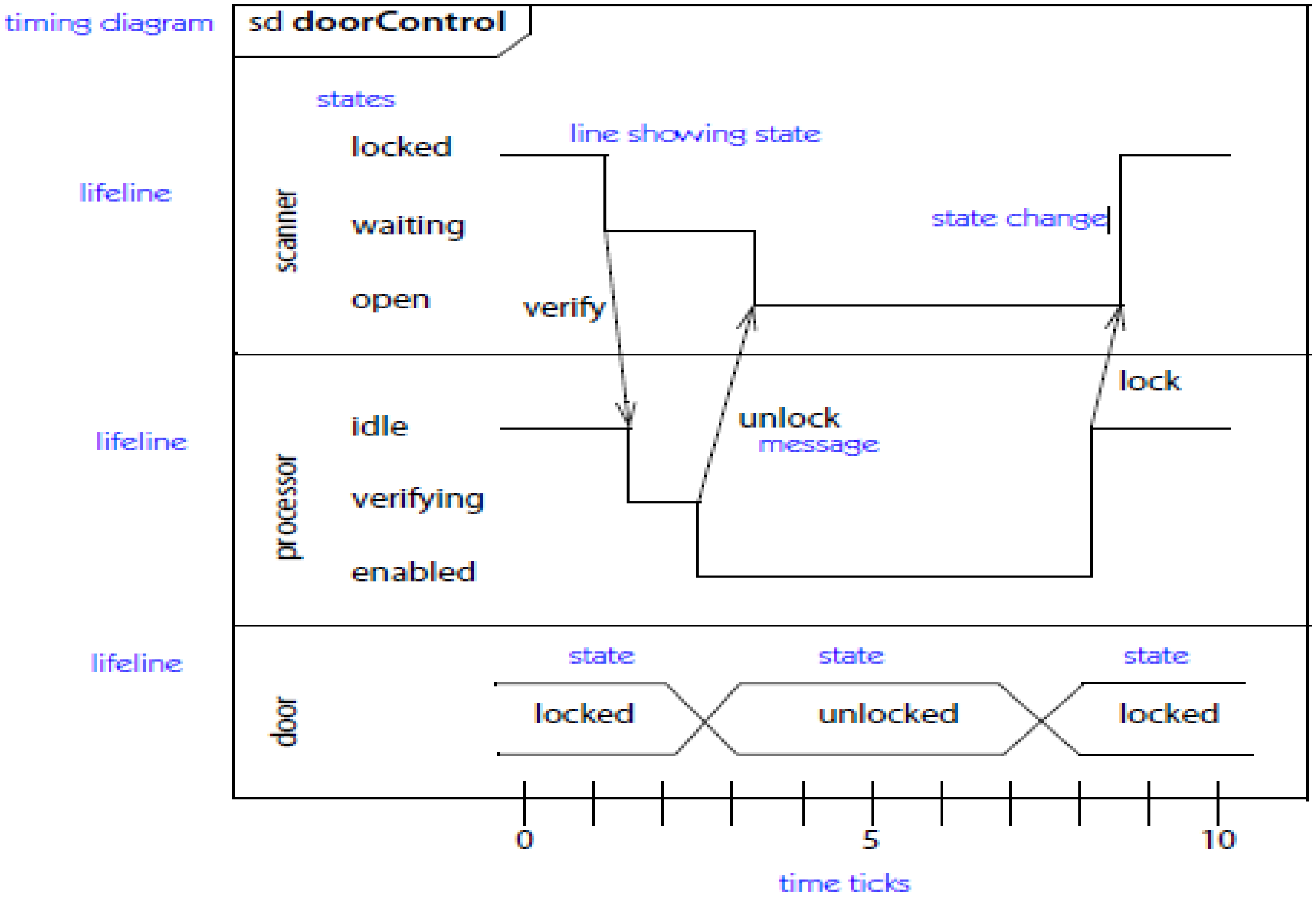
EX2 : NOTATIONS



YOUR TURN

- Door control is based on the working of processor and a scanner. Initially door is locked and scanner and processor are in idle state. When an authorized person places a finger on the sensor, it sends a message to the processor for verifying the finger print. Sensor keeps on its waiting state until the verification of unlocking is received from the processor. Verification should be done within 4 seconds after receiving message. Processor is in enabled state after verifying. Sensor remains open for 10 seconds before going into idle state. Processor state is also changed to idle state.

EX 6:


timing diagram
sd doorControl
states
line showing state
lifeline
scanner
locked
waiting
open
verify
state change
lifeline
processor
idle
verifying
enabled
unlock
message
lock
lifeline
door
state
state
state
locked
unlocked
locked
0
5
10
time ticks

That is all