description

organized software system

related systems

develop

product-line architectures

may reusable across a range of systems

*large-scale*

*reuse*

parallel other specification activities

early stage of system design

*system analysis*

system stakeholders

discussion focus

communication

control

software architecture

*model*

design

specification

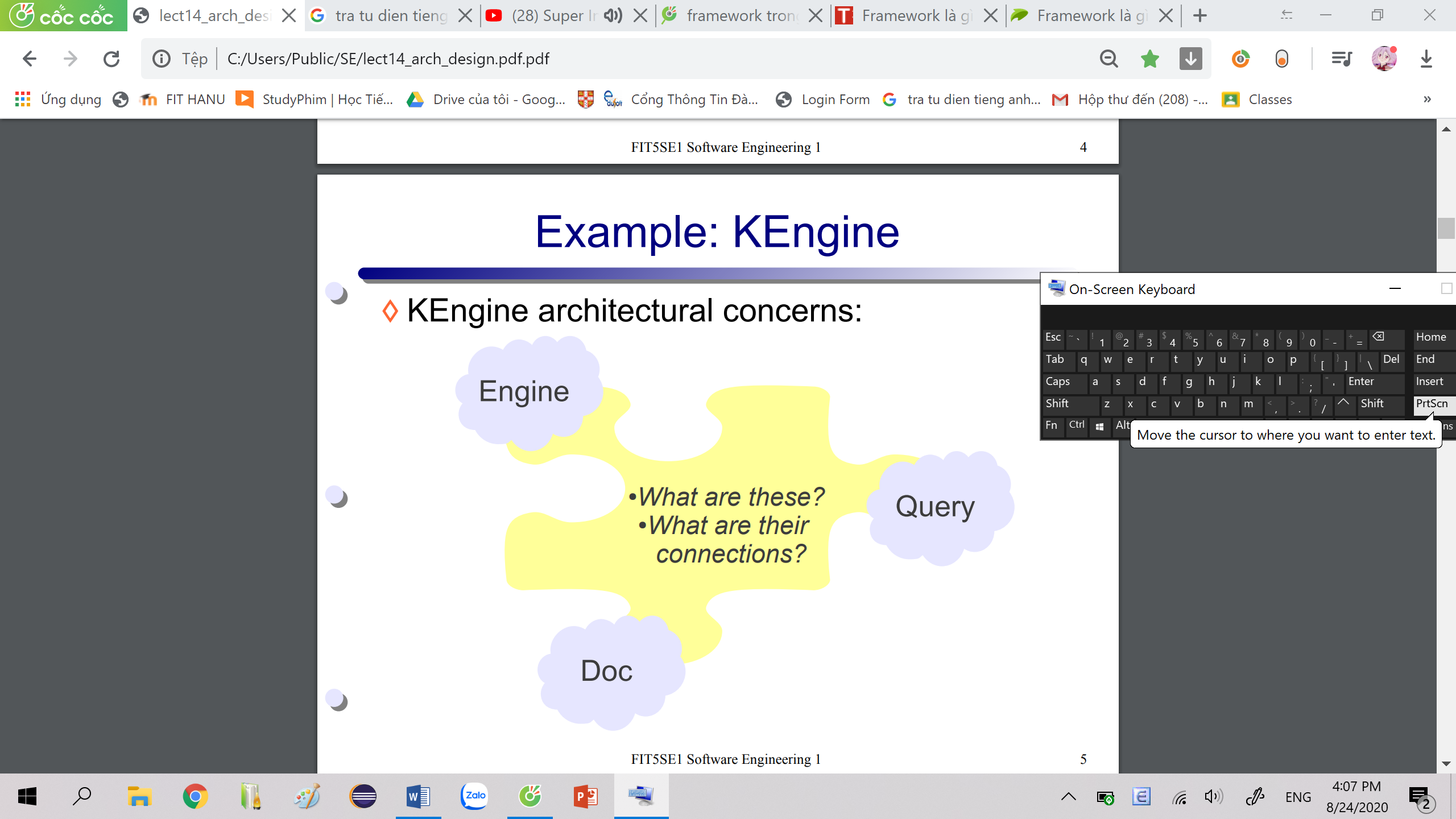
*link*

software system

identify

*design process*

Architectural modelling



KEngine architecture

KEngine

Engine

Comm

Query

Doc

WordTable

Titletable

KEngine model

document manager

use

replaceable

fine-grain

components

fault tolerance

redundant

components

mechanisims

localise

safety-critical

features

inner layers: critical assets

layered architecture

large components

communications

critical operations

use

minimise

localise

security

availability

maintainability

safety

performance

esp. class diagram

interact

depend

link 🡪 other UML models

(group) data abstractions

software components

development

component diagram

package diagram

**deployment** diagram

UML

model constructs + diagram

describe

architecture structure

language

architectural representation

Basic KEngine architecture

depends on

depends on

depends on

component

Detailed KEngine architecture

KEngine architectural model 🡪 show some components 🡪 packages

Comm

Engine

kengine

packages

TitleTable

WordTable

document manager

Query

Doc

kengine model

*distribute data + processing*

*network-based system*

data change 🡨🡪 representation

support dif pre. – same data

code complexity

3 logic components

multiple ways 🡪 interact data

unknown future requirements

***Controller****: manage users interactions*

*via keyboard, mouse, …*

***View****: manage data presentation 🡪 user*

***Model****: manage system data + operation*

***Client/ server architecture***

***MVC***

***Layer***

*grouped 🡪 layers (sub-systems)*

*identity*

architectural patterns

*stylized description* 🡪 *good design practice*

*means 🡪 reusing knowledge 🡪 generic system architectures*

distribute server – network

available shared functionality

server: failure point

unpredictable performance

increased system management overhead

data shared - range of app. / processes

load sharing 🡪 particular service

database look-up

***Client****: process 🡨 servers services*

***Network****: clients 🡨🡪 servers*

***Server****: stand-alone process*

* *specific service*

*printing, data management, …*

|  |
| --- |
| View |
| render model  request model updates  send user events 🡪 Controller |

|  |
| --- |
| Controller |
| user actions – model updates  View selection |

|  |
| --- |
| Model |
| encapsulate app. state  notify view of state change |

MVC diagramKEngine

state query

change notification

state change

user Event

select View

Query & Document

view selection

|  |
| --- |
| View |
| render document + query  Event managemnt |

|  |
| --- |
| Controller |
| network request handling  Query event handling |

Query & Document

events

|  |
| --- |
| Model |
| Document management |

Document & keyword

request

new document

notification

keyword

addition request

Client/ server architecture diagram

Bus style network

Server B

Server A

Client 3

Client 2

Client 1

**Network**

Client 3

Client 2

Client 1

Server B

Server A

3-tier architecture

Client 3

Tier 3

Data management

layer 4

Tier 2

App. logic

layer 3

Tier 1

User interface

layers 1 & 2

DBServer B

DBServer A

AppServer B

AppServer A

Client 2

Client 1

KEngine 3-tier architecture

**Internet**

Tier 2

Document & search processing logics

Tier 3

Document

management

Tier 1

Web-based

interface

AppServer

B

AppServer

A

DBServer C

DBServer B

DBServer A

Client 1

Client 2

Client 3