Research & Development the Protocol in 4G/5G

tuandn3@viettel.com.vn

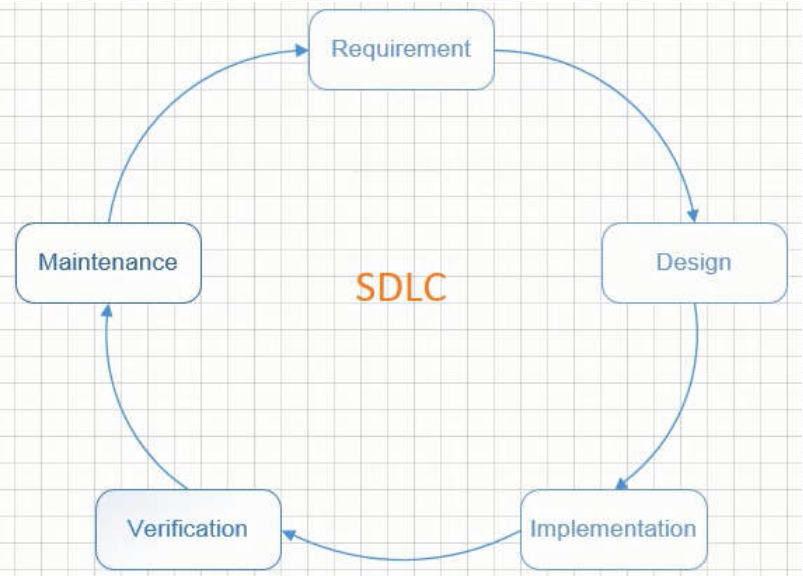
Agenda

1. Software development process
   - Waterfall model
   - eNodeB Viettel
2. Project 1 – WikiPTIT
   - Introduction
   - Practices
3. Project 2 – 4G LTE Paging simulation
   - Introduction
   - Practices
4. Project 3 – 5G NR Paging simulation
   - Introduction
   - Practices
5. Socket programming with TCP
   - Socket API
   - Example with socket API
6. Homework
   - Introduction
   - Project 2

1. Software Development Process

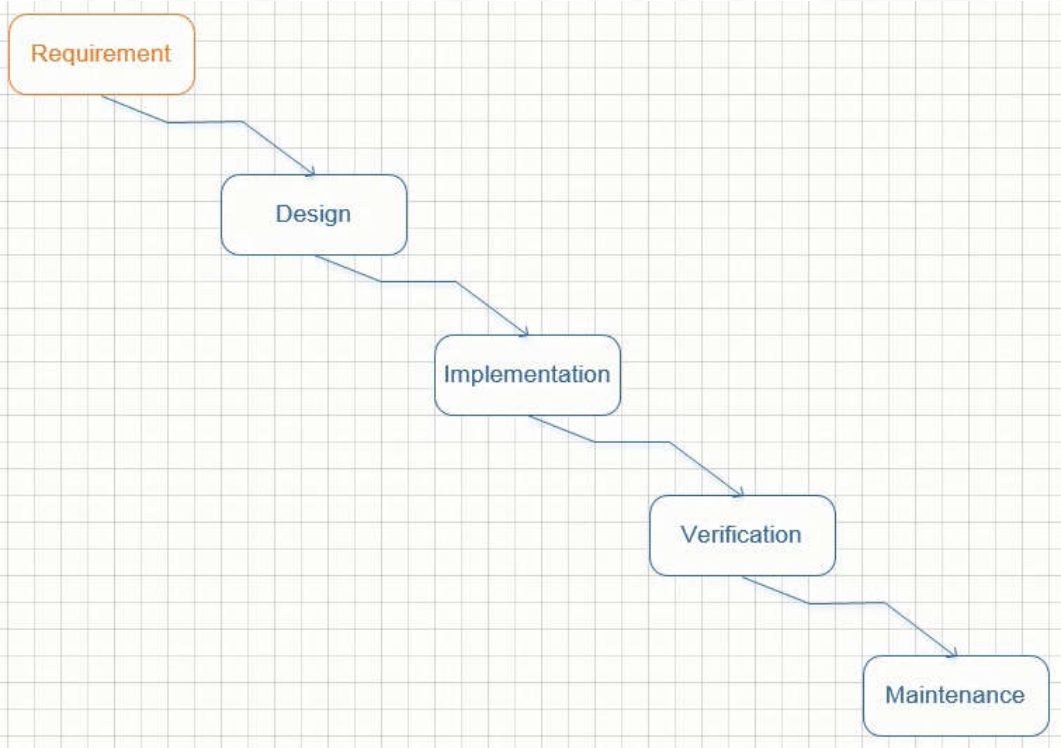
Software Development Process

- Definition
  - Software Development Process or Software development life cycle SDLC. It is a process that describes how to develop, design and maintain the software project ensuring that all the functional & user requirement, goals and objective are met. This methodology improves the quality of the software project and over all process of software development.



- Models
  - Waterfall
  - Iterative
  - Spiral
  - Agile

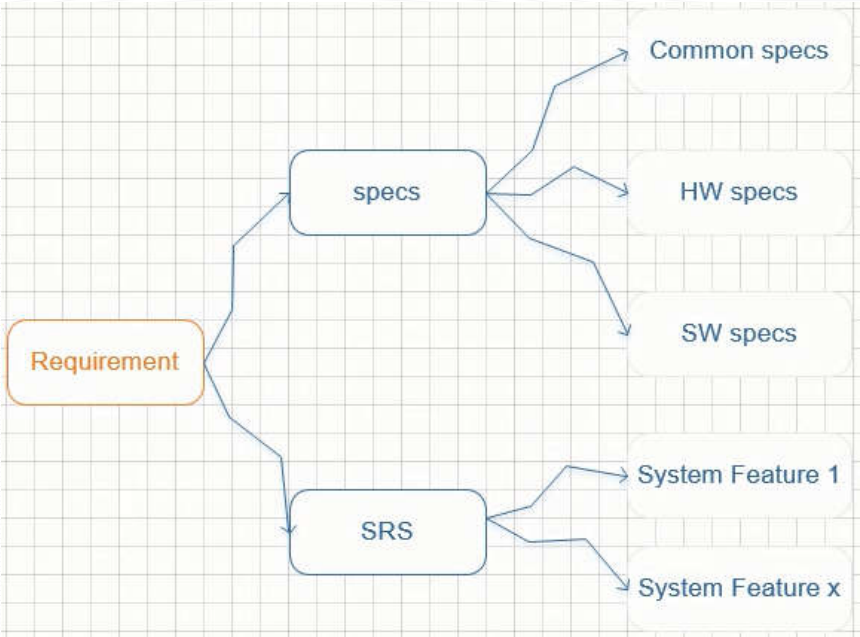
Waterfall



- Input
  - 3GPP, ITU
  - pre-study document
- Output
  - Specification
  - System Requirement Specification

Requirement – Specs

- **Purpose**: to transform the defined stakeholder requirements into a set of desired system technical requirements that will guide the design of the system



ID	eNodeB	Version 1	Description
I	**Common specs**		
1	Compliance standards	3GPP R10	
2	Duplexing mode	FDD support	
3	Bandwidth (Mhz)	20	
II	**Mechanical specs**		
1	BU weight (kg)	15	
2	RU weight (kg)	25	
3	BU H*W*D (cm)	450*350*60	
III	**Tính năng trên gNodeB (NSA)**		
1	Paging	x	
2	RRC Connection Establishment	x	
3	RRC Connection Release	x	

ID	gNodeB	Version 1	Description
I	**Common specs**		
1	Compliance standards	3GPP R15	
2	Duplexing mode	TDD support	
3	Bandwidth (Mhz)	100	
II	**Mechanical specs**		
1	BU weight (kg)	10	
2	RU weight (kg)	20	
3	BU H*W*D (cm)	400*300*50	
III	**System Feature on NSA mode**		
1	Paging	x	
2	SgNB Addition Procedure	x	
3	SgNB Release procedure	x	

Requirement – SRS

Table of Contents

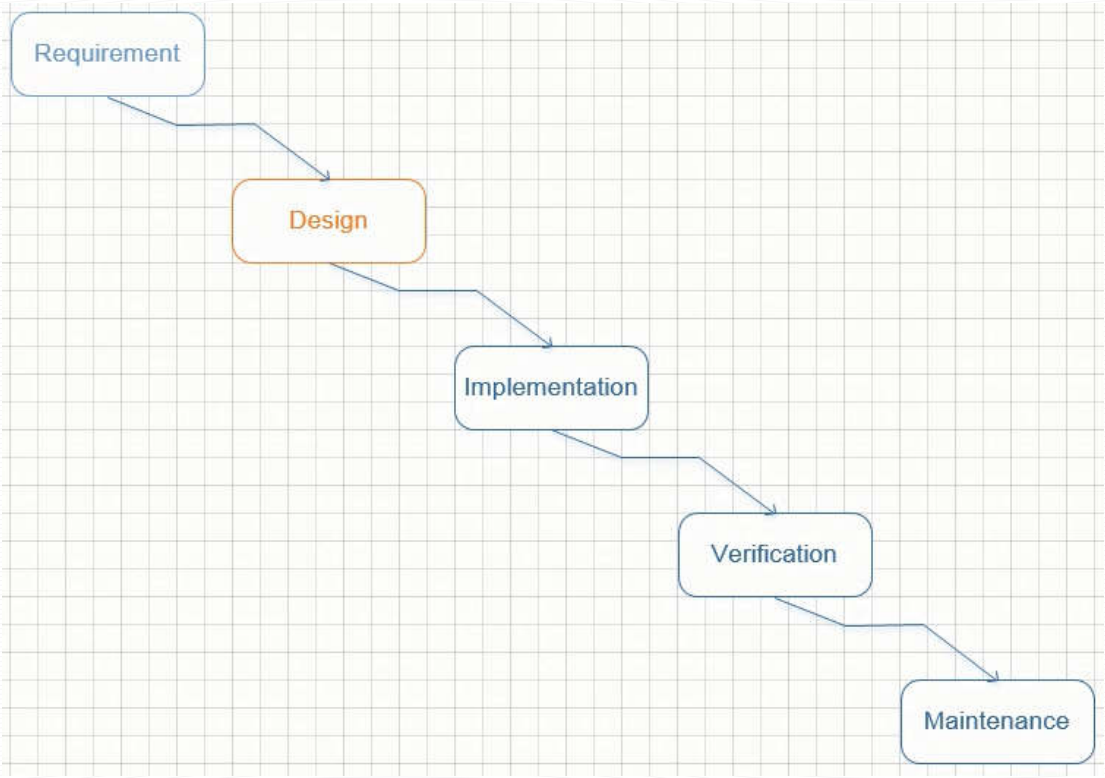
1. Introduction
   - 1.1 Purpose
   - 1.2 Scope
   - 1.3 Definitions, acronyms, and abbreviations
   - 1.4 References
   - 1.5 Overview
2. Overall description
   - 2.1 Product perspective
   - 2.2 Product functions
   - 2.3 User characteristics
   - 2.4 Constraints
   - 2.5 Assumptions and dependencies
3. Specific requirements (See 5.3.1 through 5.3.8 specific requirements. See also Annex A for se this section of the SRS.)

Appendixes

3. Specific requirements
- 3.1 External interface requirements
  - 3.1.1 User interfaces
  - 3.1.2 Hardware interfaces
  - 3.1.3 Software interfaces
  - 3.1.4 Communications interfaces
- 3.2 System features
  - 3.2.1 System Feature 1
    - 3.2.1.1 Introduction/Purpose of feature
    - 3.2.1.2 Stimulus/Response sequence
    - 3.2.1.3 Associated functional requirements
      - 3.2.1.3.1 Functional requirement 1
      - .
      - .
      - .
      - 3.2.1.3.*n* Functional requirement *n*
  - 3.2.2 System feature 2
  - .
  - .
  - .
  - 3.2.*m* System feature *m*
  - .
  - .
  - .
- 3.3 Performance requirements
- 3.4 Design constraints
- 3.5 Software system attributes
- 3.6 Other requirements

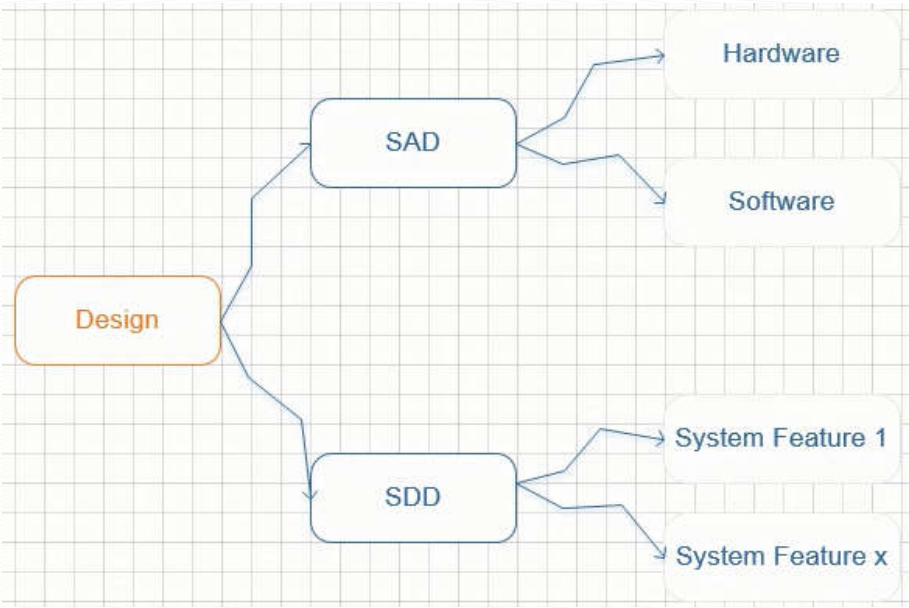
- 1.2 Phạm vi
- 1.3 Tổng quan
- 2 mô tả chung hệ thống
  - 2.1 Mô hình tổng quan
- 3 yêu cầu chức năng
  - 3.1 System Feature 1
    - 3.1.1 Mục đích
    - 3.1.2 Luồng xử lý
    - 3.1.3 Chức năng liên quan
      - 3.1.3.1 Yêu cầu chức năng 1
        - 3.1.3.1.1 Mô tả
        - 3.1.3.1.2 Đầu vào
        - 3.1.3.1.3 Quy trình
        - 3.1.3.1.4 Đầu ra
      - 3.1.3.2 Yêu cầu chức năng N
- 4 YÊU CẦU GIAO TIẾP HỆ THỐNG NGOÀI
  - 4.1 Yêu cầu giao diện
  - 4.2 Giao tiếp phần cứng
  - 4.3 Giao tiếp phần mềm
  - 4.4 Truyền tải thông tin
- 5 yêu cầu phi chức năng
  - 5.1 Hiệu suất hệ thống
  - 5.2 An toàn hệ thống
  - 5.3 Bảo mật hệ thống
- 6 yêu cầu khác
- 7 PHỤ LỤC
  - 7.1 Tài liệu tham khảo

Design



- Input
  - System Specification
  - System Requirement Specification
- Output
  - System Architecture Design
  - System Detailed Design

Design

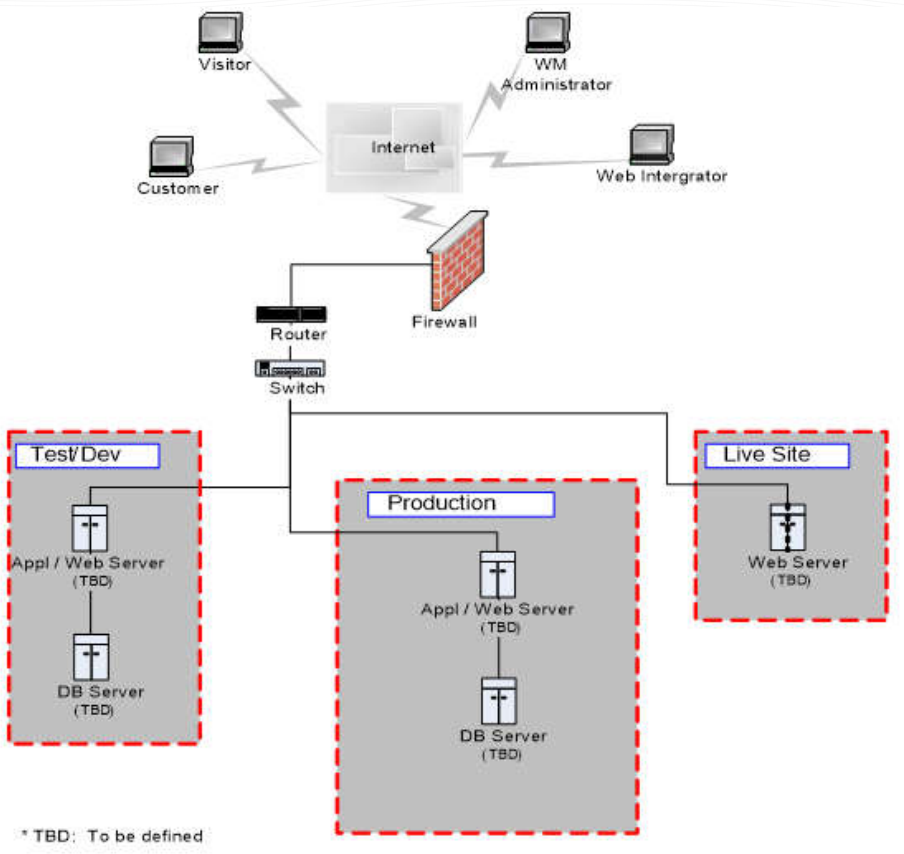


- **Architecture Design**: The process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system.

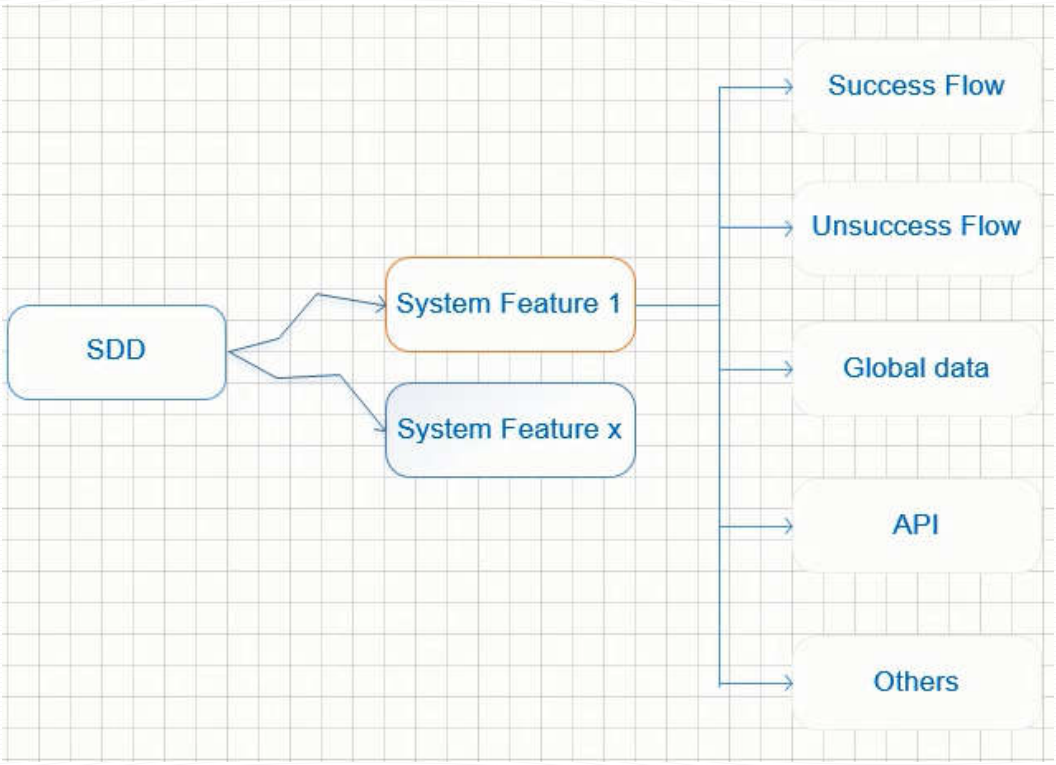
- **Detailed Design**: The process of refining and expanding the Architecture Design of a system or component to the extent that the design is sufficiently complete to be implemented

Design – Architecture Design

- 1 GIỚI THIỆU
  - 1.1 Mục đích
  - 1.2 Phạm vi
  - 1.3 Tổng quan
- 2 MÔ HÌNH TỔNG QUAN
- 3 MỤC TIÊU thiết kế & RÀNG BUỘC
  - 3.1 Mục tiêu
  - 3.2 Ràng buộc
- 4 KIẾN TRÚC PLATFORM
  - 4.1 Tổng quan kiến trúc Platform
  - 4.2 Kiến trúc phần cứng
  - 4.3 Kiến trúc phần mềm
- 5 KIẾN TRÚC HỆ THỐNG
  - 5.1 Layer 1
  - 5.2 Layer N
- 6 MÔ HÌNH THÀNH PHẦN
- 7 CÁC THÀNH PHẦN CHÍNH HỆ THỐNG
- 8 THÀNH PHẦN HỖ TRỢ YÊU CẦU PHI CHỨC NĂNG
  - 8.1 Hiệu năng
  - 8.2 Bảo mật
  - 8.3 Đóng gói
- 9 MÔ HÌNH TRIỂN KHAI
- 10 GIẢ THIẾT VÀ RÀNG BUỘC
- 11 PHỤ LỤC
  - 11.1 Tài liệu tham khảo

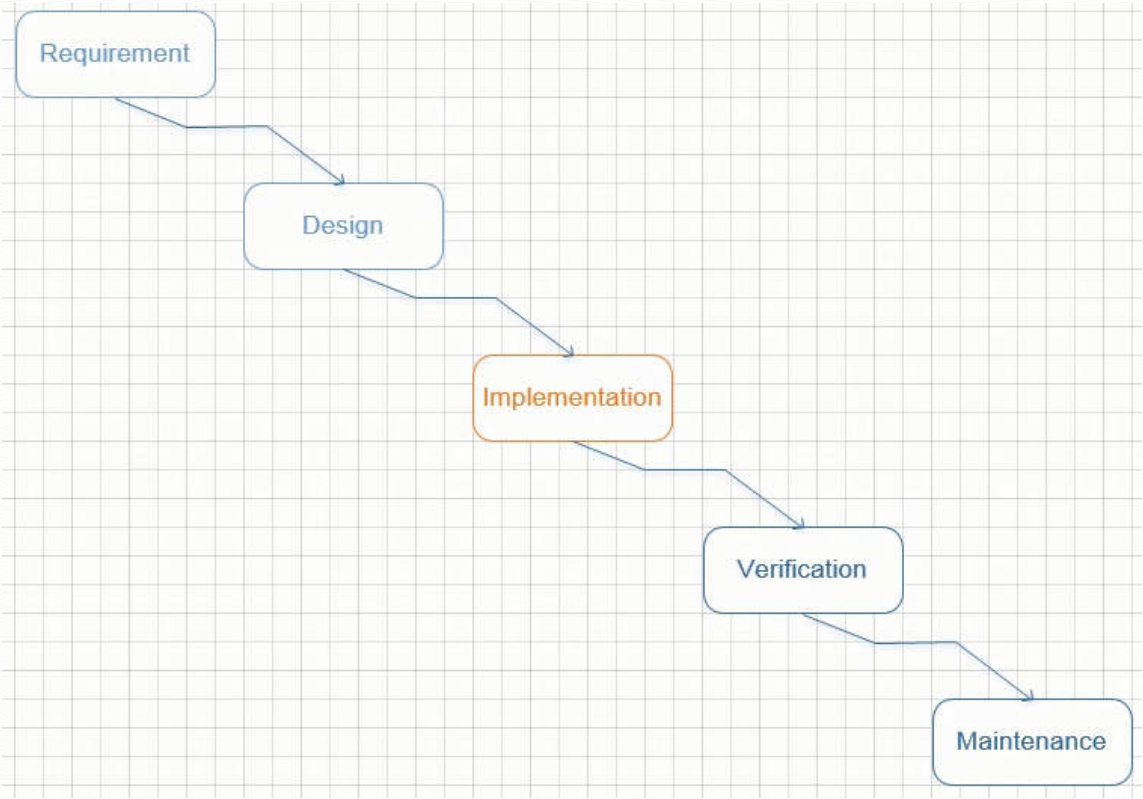


Design – Detailed Design



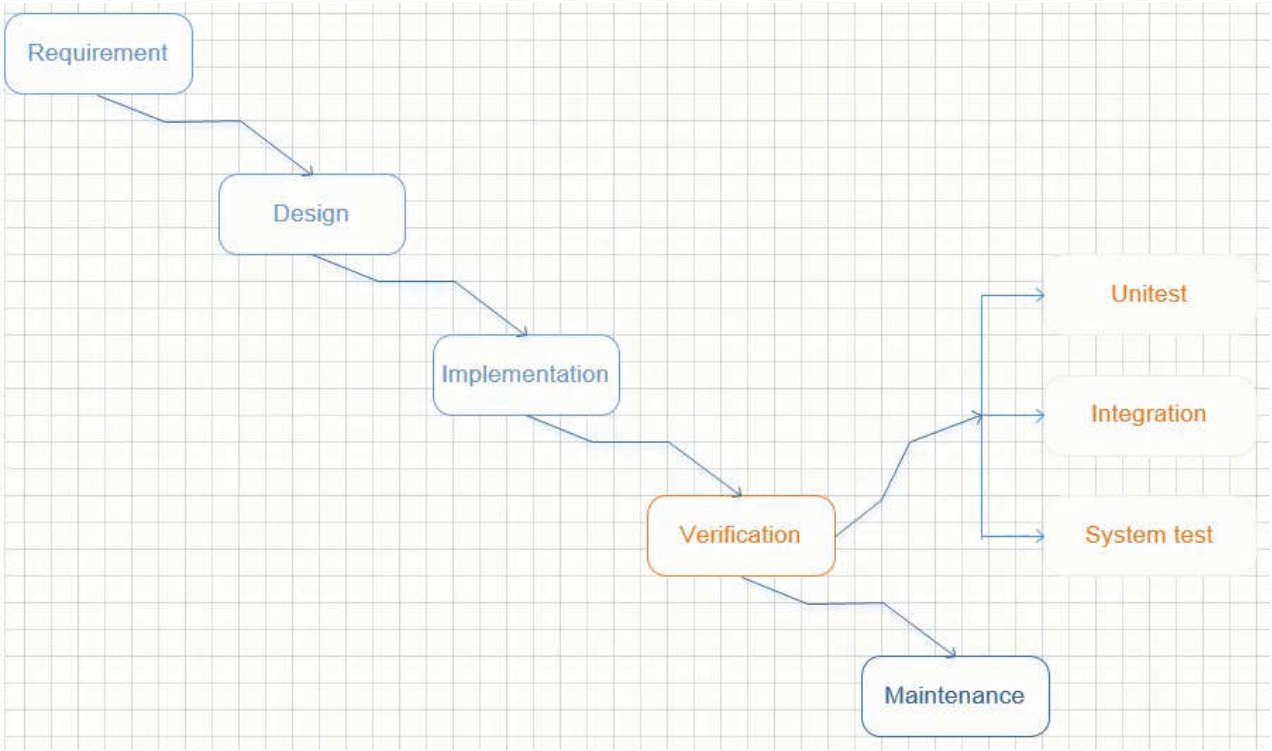
- 1. GIỚI THIỆU
  - 1.1 Mục đích
  - 1.2 Phạm vi
  - 1.3 Thuật ngữ
  - 1.4 Tài liệu tham khảo
- 2. TỔNG QUAN HỆ THỐNG
  - 2.1 Tổng quan kiến trúc phần mềm
  - 2.2 Tổng quan kiến trúc phần cứng (Tùy chọn)
- 3. THIẾT KẾ MODULE PHẦN MỀM
  - 3.1 SDD001_RRC_PAGING
    - 3.1.1 Tổng quan thủ tục
    - 3.1.2 Luồng xử lý thành công chi tiết
      - 3.1.2.1 Tổng quan luồng xử lý thành công
      - 3.1.2.2 CSC nhận được bản tin S1AP_PAGING_IND
      - 3.1.2.3 CSC thực hiện thủ tục SFN_TIMER expired
      - 3.1.2.4 SFN cập nhật do MAC báo lên khi xảy ra mất đồng bộ
      - 3.1.2.5 SFN đồng bộ theo chu kỳ
    - 3.1.3 Luồng xử lý thất bại chi tiết.
    - 3.1.4 Module
      - 3.1.4.1 Global Context
- 4. THIẾT KẾ DATA
  - 4.1 Dữ liệu tĩnh
  - 4.2 Kiểu dữ liệu
  - 4.3 Dữ liệu động (Dynamic Data)

Implementation



- Input
  - System Architecture Design
  - System Detailed Design
  - Coding convention
- Output
  - Source code, configuration
  - Runtime package

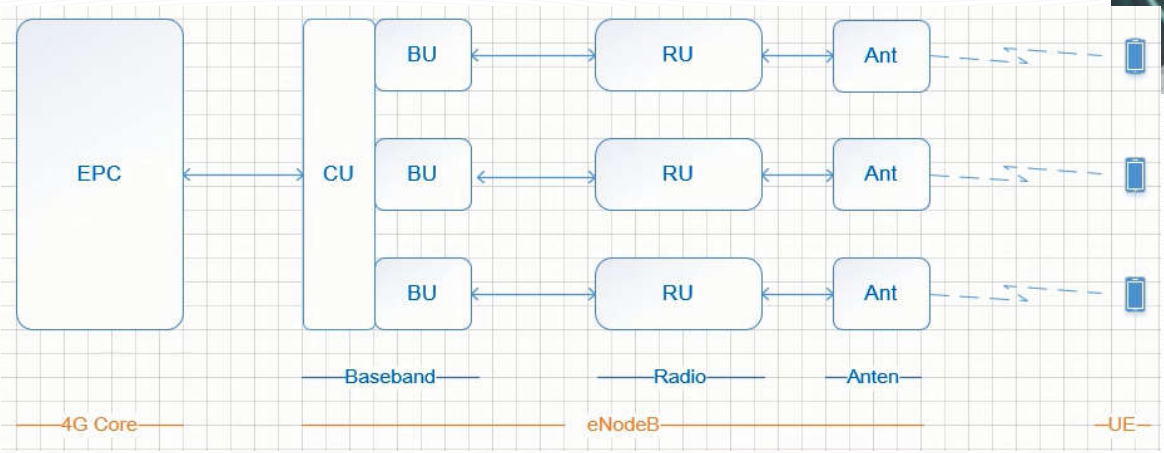
Verification



- Input
  - Runtime package
  - Test case
- Output
  - Test report
  - Bug list

eNodeB Viettel

- Baseband Block
  - CU: Control Unit
  - BU: Baseband unit
- Radio Block
  - RU: Radio Unit
  - Ant: Anten
- 4G Core
  - MME: Mobility Management Entity
  - SGW/ PGW: Serving Gateway/ Packet Data Gateway



eNodeB Viettel – specification

Technical Specifications

Item	vBBU - 366b	vRRU - 1842b/2642b	vRRU - 1844b/2644b	SMC - 1812b/2612b
Compliance standards	3GPP Release 10			
Duplexing mode	FDD			
Supported Bandwidth (MHz)	1.4, 3, 5, 10, 15, 20 Mhz			
Bands		1800/2600Mhz	1800/2600Mhz	1800/2600Mhz
Maximum of Cell Carriers per BBU	6			
Maximum of Cell Carriers per RRU		1	2	1
Transmit and Receive Channels		2T2R	4T4R	2T2R
Maximum Output Power		2x40W	4x40W	2x10W
IBW (Instantaneous BandWidth)		20	40	20
Receiver Sensitivity (two ways)		-109dBm	-109dBm	-109dBm
Maximum RRC-connected users per Cell	1200			
Maximum RRC-connected users per BBU	3600			
Maximum DL/UL Throughput per Cell (Mbps)	300/75			
Maximum DL/UL Throughput per BBU (Mbps)	900/225			
Call Attempts per Second	70 CAPS			

Mechanical specifications

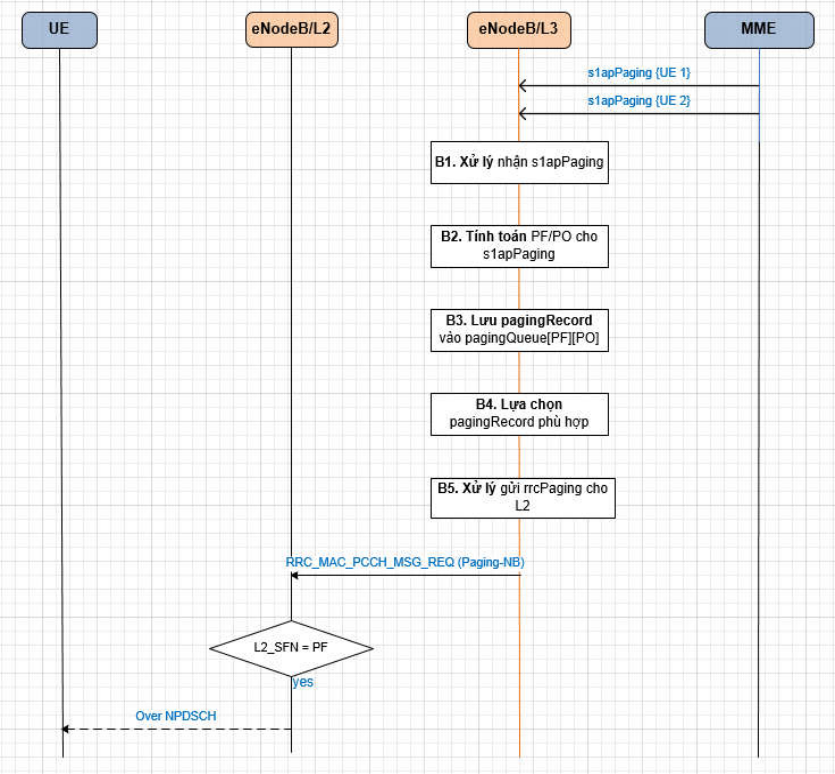
Item	vBBU-366b	vRRU - 1842b/2642b	vRRU - 1844b/2644b	SMC - 1812b/2612b
HxWxD	446x344x59mm;	320x400x135 mm	500x350x147 mm	455x290x164 mm
Weight	9Kg	16 Kg	25 Kg	15 Kg
Mount	Pole	Pole	Pole	Pole, Wall, Ceiling

Environment specifications

Item	vBBU-366b	vRRU - 1842b/2642b	vRRU - 1844b/2644b	SMC - 1812b/2612b
Operating temperature	-10°C to +55°C	-10°C to +55°C	-10°C to +55°C	-10°C to +55°C
Relative Humidity	5% to 95%	5% to 95%	5% to 95%	5% to 95%
Absolute Humidity	1 to 30 g/m3	1 to 30 g/m3	1 to 30 g/m3	1 to 30 g/m3
Ingress Protection Type	IP20	IP67	IP67	IP65

eNodeB – SRS Paging Feature

- 1 GIỚI THIỆU
  - 1.1 Mục đích
  - 1.2 Phạm vi
  - 1.3 Thuật ngữ
  - 1.4 Tài liệu tham khảo
  - 1.5 Quy định về ký hiệu
- 2 MÔ TẢ HỆ THỐNG
  - 2.1 Thông tin chung
  - 2.2 Các yêu cầu của hệ thống
  - 2.3 Sự hạn chế và giả định, phụ thuộc
- 3 ĐẶC TẢ YÊU CẦU
  - 3.1 Chồng giao thức eNodeB cho tính năng Paging eDRX
    - 3.1.1 Mô tả chung NB-IoT Paging eDRX
    - 3.1.2 Luồng NB-IoT Paging eDRX tại eNodeB
    - 3.1.3 S1AP Paging
    - 3.1.4 RRC Paging
    - 3.1.5 Quản lý SFN và H-SFN
    - 3.1.6 Tính toán Paging Frame và Paging Occasion
    - 3.1.7 Tính toán PH, PTW_start và PTW_end
    - 3.1.8 Tính toán Paging Repetition
  - 3.2 Các tham số
  - 3.3 Các counter



eNodeB – Verification

- Tools
  - EPC emulator: Polaris
  - UE emulator: Aeroflex
  - UE 4G:
  - UE analyst tool
    - QXDM/QCAT by Qualcom
    - Nemo by Keysight
    - Tems by infovista

Time	Type	Description	Subtitle	recti
2018 Ja...	0xB112	Reserved		
2018 Ja...	0xB067	LTE Mac UL Tx Statistics		
2018 Ja...	0xB16B	LTE PDCCH-PHICH Indication ...		
2018 Ja...	0xB172	LTE Uplink PKT Build Indication		
2018 Ja...	0xB0A4	LTE PDCP DL Statistics Pkt		
2018 Ja...	0xB114	LTE LL1 Serving Cell Frame Tim...		
2018 Ja...	0xB16B	LTE PDCCH-PHICH Indication ...		
2018 Ja...	0xB14D	LTE LL1 PUCCH CSF		
2018 Ja...	0xB193	LTE ML1 Serving Cell Meas Res...		
2018 Ja...	0xB16B	LTE PDCCH-PHICH Indication ...		
2018 Ja...	0xB0C0	LTE RRC OTA Packet	PCCH / Paging	BS...
2018 Ja...	0xB193	LTE ML1 Serving Cell Meas Res...		
2018 Ja...	0xB132	LTE LL1 PDSCH Decoding Results		
2018 Ja...	0xB16F	LTE PUCCH Power Control		
2018 Ja...	0xB14D	LTE LL1 PUCCH CSF		
2018 Ja...	0xB16B	LTE PDCCH-PHICH Indication ...		
2018 Ja...	0xB112	Reserved		
2018 Ja...	0xB112	Reserved		
2018 Ja...	0xB172	LTE Uplink PKT Build Indication		
2018 Ja...	0xB112	Reserved		
2018 Ja...	0xB112	Reserved		

```
2018 Jan 22  08:37:35.718  [EF]  0xB0C0  LTE RRC OTA Packet  --  PCCH / Paging
Pkt Version = 7
RRC Release Number.Major.minor = 10.7.1
Radio Bearer ID = 0, Physical Cell ID = 310
Freq = 1750
SysFrameNum = 31, SubFrameNum = 9
PDU Number = PCCH Message,    Msg Length = 7
SIB Mask in SI =  0x00

Interpreted PDU:

value PCCH-Message ::=
{
  message c1 : paging :
      {
        pagingRecordList
        {
          {
            ue-Identity s-TMSI :
              {
                mmec '00001000'B,
                m-TMSI '11100001 11100011 11000000 01001010'B
              },
            cn-Domain ps
          }
        }
      }
}

          Length:  36
          Header:  24 00 C0 B0 EF 11 1E 99 3C A3 DF 00
          Payload: 07 0A 71 00 36 01 D6 06 F9 01 04 00
                   00 00 00 07 00 40 00 8E 1E 3C 04 A0
```

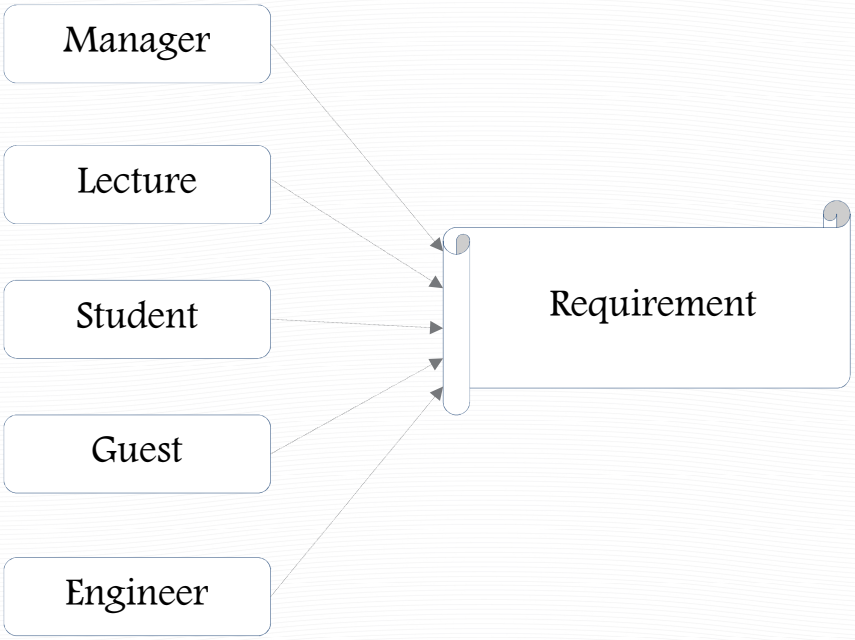
2. Project 1 – WikiPTIT

WikiSET

- Mission: Build the online WikiPTIT which student, lecturer can search thesis, paper, course.
- Practices:
  - Task 1: gather requirements
  - Task 2: build the specs
  - Task 3: make the System Feature x
  - Task 4: make the Architecture Design
  - Task 5: make the Detailed Design
  - Task 6: Implementation & Verification

Task 1: gather requirement

- **Purpose**: to transform the defined stakeholder requirements into a set of desired system technical requirements that will guide the design of the system



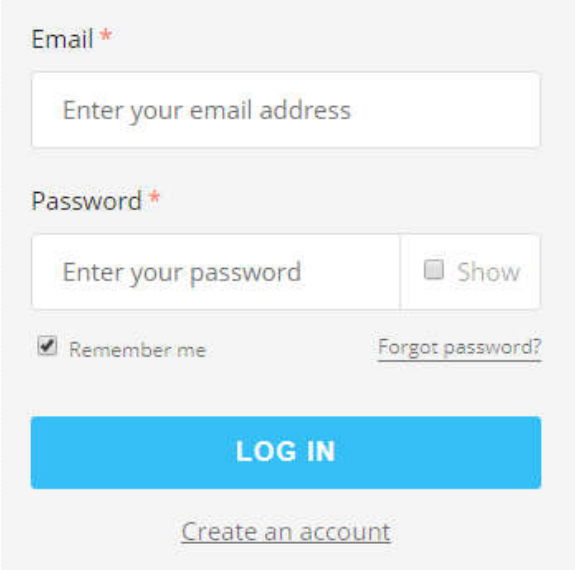
Task 2: build the specs

- Common specs
- Performance specs
- System Feature specs

ID	eNodeB	Version 1	Version 2	Description
I	**Common specs**			
1	Linux server CentOS 8		x	
2	Database MySQL 8.0		x	
3				
II	**Performance specs**			
1	Max connections		1000	
2	Max active connection		100	
3	Response time (ms)		100	
III	**System Feature Specs**			
1	account management		x	
2	student login management		x	
3	lecture login management		x	
4	guest login management		x	
5	course search		x	
6	thesis search		x	

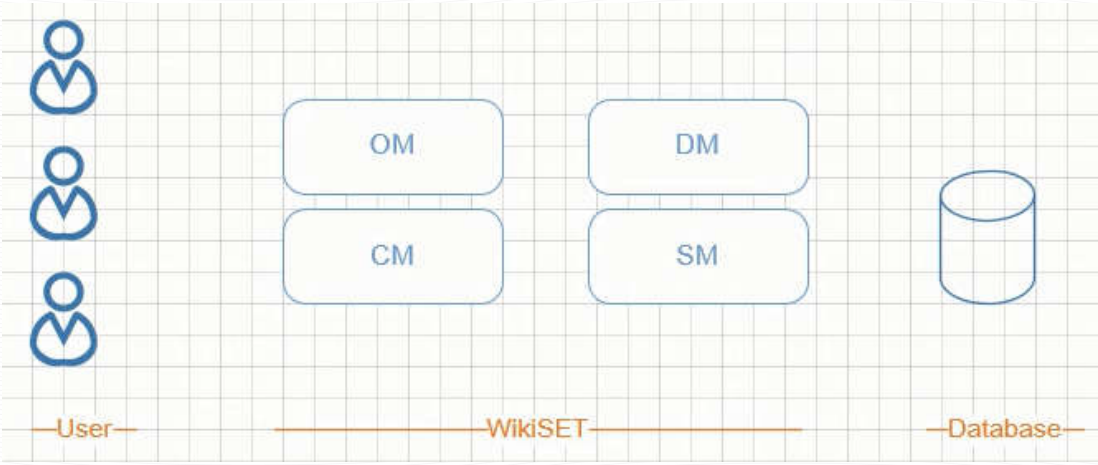
Task 3: make the System Requirement Feature X

- Student login management
  - Purpose
  - Flow
  - Functions
    - Login success
    - Login failure
    - History, log
    - View status
    - Support BachKhoa account

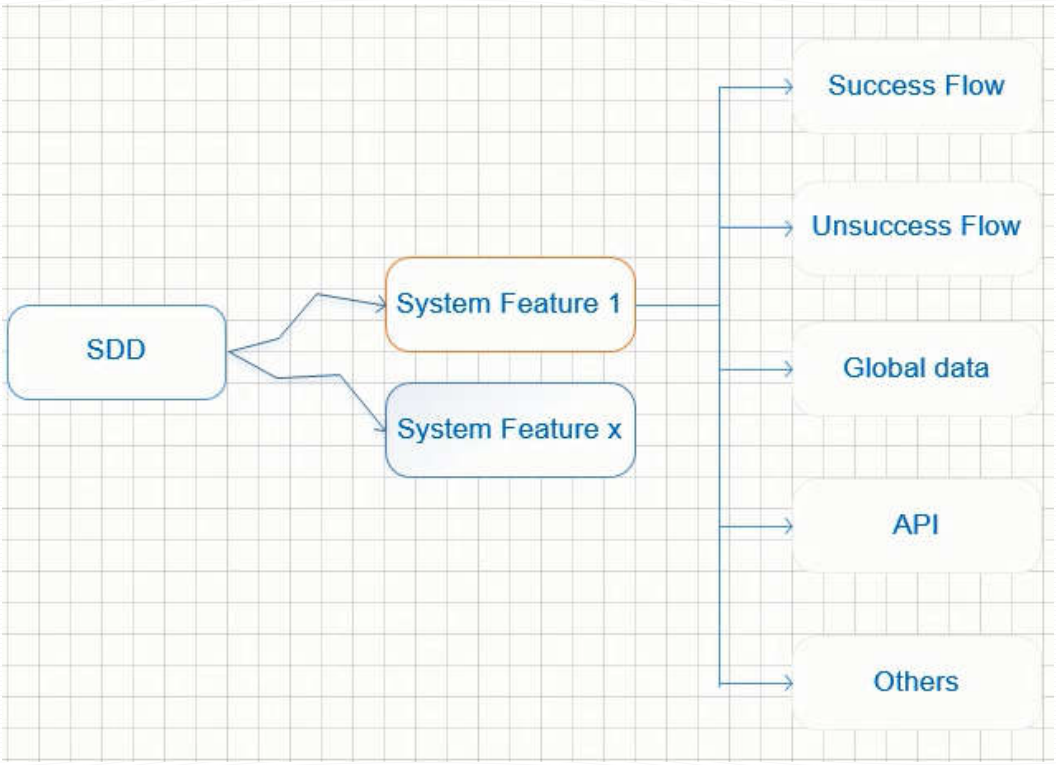


Task 4: make the Architecture Design

- Database
- Server
- Modules
  - Database Management
  - Connection Management
  - Search Management
  - Operation Management



Task 5: make the Detailed Design



- Student login management
  - Success flow
  - Unsuccessful Flow
  - Global data
  - API
  - State machine

Task 6: Implementation & Verification

- Implementation
  - Tools
  - WikiPTIT
  - Client simulation
- Verification
  - Unitest
  - Integration
  - System

3. Project 2 – 4G LTE Paging simulation

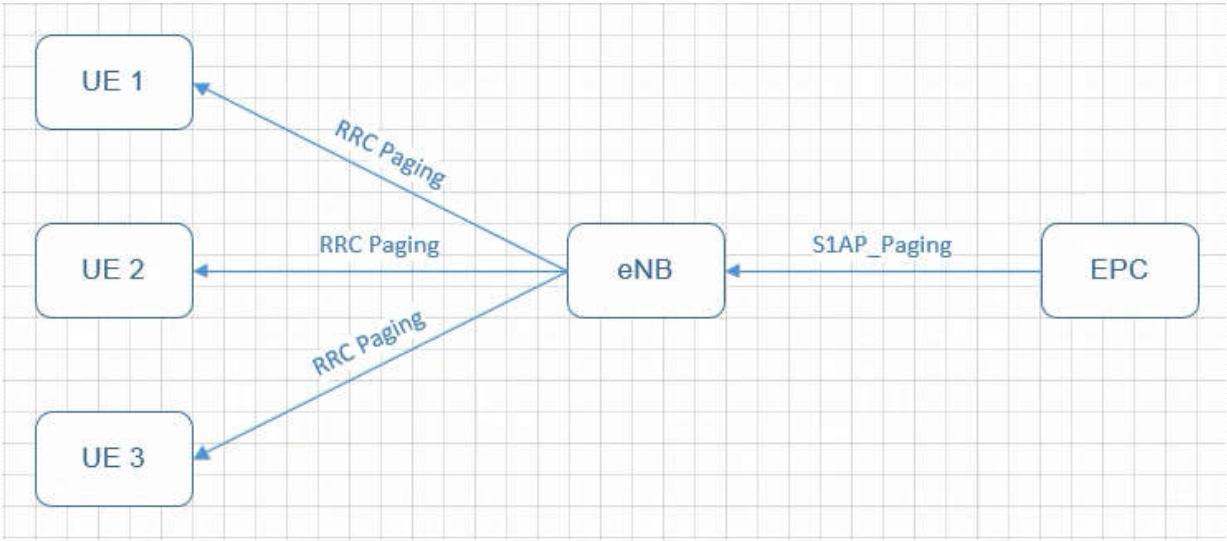
4G LTE Paging simulation

- Mission: Simulate the 4G LTE Paging system
- Practices:
  - Task 1: gather requirement
  - Task 2: build the specs
  - Task 3: make the System Paging x
  - Task 4: make the Architecture Design
  - Task 5: make the Detailed Design
  - Task 6: Implementation & Verification

3GPP reference

- https://www.3gpp.org/ftp/Specs/archive/36_series (release 10)
- 3GPP TS 36.331: "Evolved Universal Terrestrial Radio Access (E-UTRAN); Radio Resource Control (RRC) Protocol Specification".
- 3GPP TS 36.304: "Evolved Universal Terrestrial Radio Access (E-UTRA), User Equipment (UE) procedures in idle mode".
- 3GPP TS 36.413: "Evolved Universal Terrestrial Radio Access (E-UTRAN); S1 Application Protocol (S1AP)".

Pre-study



- Process
  - EPC
  - eNB
  - UE
- IPC
  - TCP socket
- 3GPP Paging procedure
  - UE Paging procedure
  - RRC Paging procedure
  - S1AP paging procedure

4. Project 3 – 5G NR Paging simulation

5G NR Paging simulation

- Mission: Simulate the 5G NR Paging system
- Practices:
  - Task 1: gather requirement
  - Task 2: build the specs
  - Task 3: make the System Paging x
  - Task 4: make the Architecture Design
  - Task 5: make the Detailed Design
  - Task 6: Implementation & Verification

3GPP reference

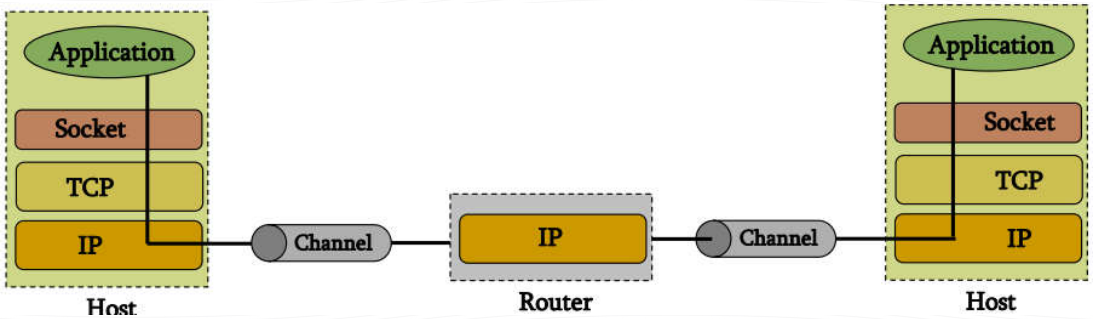
- https://www.3gpp.org/ftp/Specs/archive/38_series (release 15)
- 3GPP TS 36.413: "Evolved Universal Terrestrial Radio Access Network (E-UTRAN); S1 Application Protocol (S1AP)".
- 3GPP TS 38.304: "NR; User Equipment (UE) procedures in idle mode and in RRC inactive state".
- 3GPP TS 38.331: "NG-RAN; Radio Resource Control (RRC) Protocol Specification".
- 3GPP TS 38.401: "NG-RAN; Architecture description".
- 3GPP TS 38.410: "NG-RAN; NG general aspects and principles".

5. Socket programming with TCP

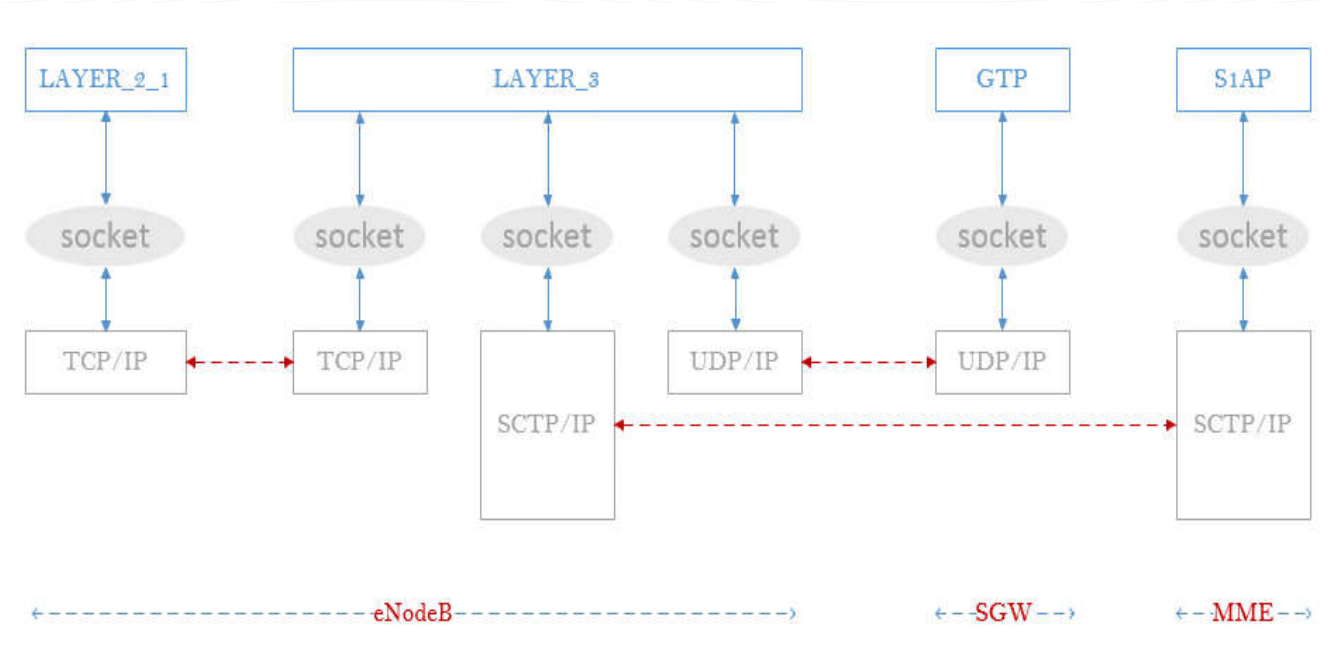
Socket

❖ Introduction

- Socket = IP address + Port
- Socket API or Berkeley socket was mostly written by University of California, Berkeley
- Socket API has used in many OS
- Application call this socket API to do the inter-process communication (same host or different node)
- Socket API provide some types of protocol sockets as TCP socket, UDP socket, SCTP socket

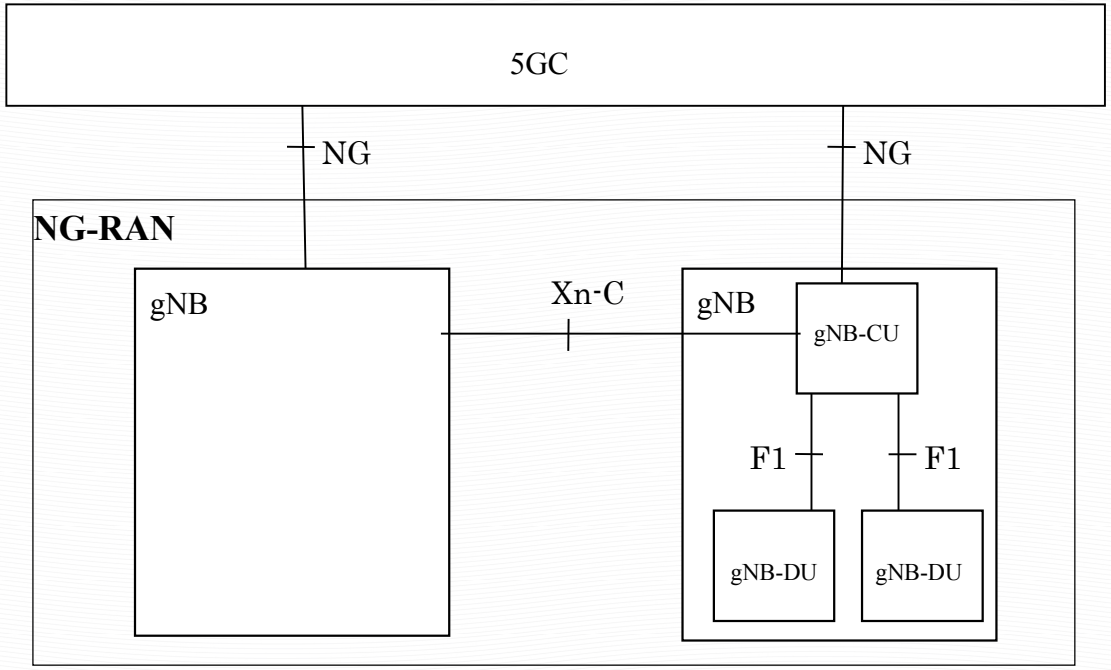


eNodeB Viettel & Socket



gNodeB Viettel & Socket

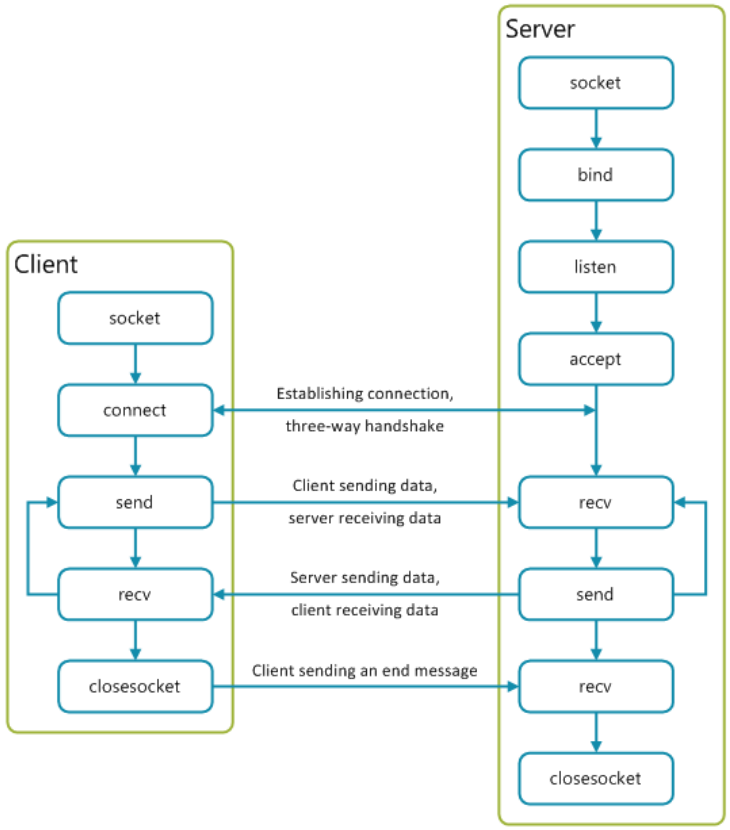
- 3GPP release 15
- Protocol stack



TCP Client – Server

Client-Server communication

- Server
  - passively waits for and responds to clients
  - passive socket
- Client
  - initiates the communication
  - must know the address and the port of the server
  - active socket



Flow Diagram for BSD Sockets Communication using TCP

Tutorial by Eleftherios Kosmas

socket

```
#include <sys/socket.h>

int socket (int family, int type, int protocol);

/* Returns: non-negative descriptor if OK, -1 on error */
```

- *socket()*: system call creates a new socket

- Family or Domain
  - AF_UNIX (same host)
  - AF_INET (via IPv4)
  - AF_INET6 (via IPv6)

- Family or Domain
  - SOCK_STREAM : reliable, bidirectional, byte-stream
  - SOCK_DGRAM: unreliable, connectionless,

- protocol
  - TCP use with SOCK_STREAM
  - UDP use with SOCK_DGRAM
  - SCTP

connect

```
#include <sys/socket.h>

int connect(int sockfd, const struct sockaddr *servaddr, socklen_t addrlen);

/* Returns: 0 if OK, -1 on error */
```

- Bind
  - The *sockfd*: file descriptor obtain from a previous call to *socket()*
  - The *myaddr*: a pointer to a structure specifying the address
  - The *addrlen*: specifies the size of the address structure.

- *socket()*: system call establishes a connection with another socket

bind

```
#include <sys/socket.h>

int bind (int sockfd, const struct sockaddr *myaddr, socklen_t addrlen);

/* Returns: 0 if OK,-1 on error */
```

- *Bind()*: system call binds a socket to an address.

- Bind
  - The *sockfd*: file descriptor obtain from a previous call to *socket()*
  - The *myaddr*: a pointer to a structure specifying the address
  - The *addrlen*: specifies the size of the address structure.

```
struct sockaddr {
    sa_family_t sa_family;          /* Address family (AF_* constant) */
    char        sa_data[14];        /* Socket address (size varies
                                       according to socket domain) */
};
```

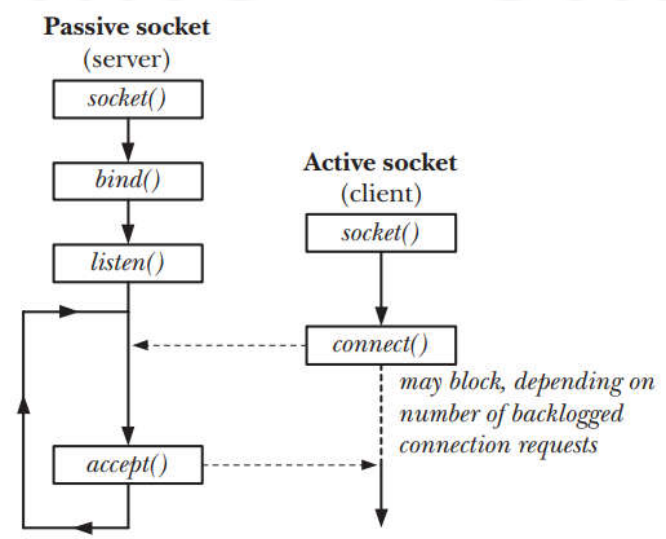
listen

```
#include <sys/types.h>          /* See NOTES */
#include <sys/socket.h>

int listen(int sockfd, int backlog);
```

- listen
  - Return 0 on success, or –1 on error
  - The *sockfd*: file descriptor obtain from a previous call to *socket()*
  - The *backlog*: argument allow us to limit the number of such pending connection

- *listen()*: system call allow a stream socket to accept incoming connections from other sockets.



accept

```
#include <sys/socket.h>

int accept (int sockfd, struct sockaddr *cliaddr, socklen_t *addrlen);

/* Returns: non-negative descriptor if OK, -1 on error */
```

- *accept()*: system call accepts a connection from a peer application on a listening stream socket, and optionally returns the address of the peer socket.

- Bind
  - The *sockfd*: file descriptor obtain from a previous call to *socket()*
  - The *cliaddr*: a pointer to a structure specifying the client address
  - The *addrlen*: specifies the size of the address structure.

write or send

```
#include <unistd.h>

ssize_t write(int fd, const void *buf, size_t count);
```

- sockfd
- sockaddr
- addrlen

read or recv

```
#include <unistd.h>

ssize_t read(int fd, void *buf, size_t count);
```

- fd
- buf
- count

6. Project 3 – 5G NR Paging simulation

4G LTE Paging simulation

- Project 2
- Report: pdf, 3-5 pages
- Team: 3 people