<디지털 법정 시스템>

**Software Architecture Description**

작성자: 최호석

(소속: VD 사업부)

Revision History

|  |  |  |
| --- | --- | --- |
| Version | Date | Summary |
| 1  (Interim) | 2024/04/24 | Project Overview 작성  System Overview 작성  Architectural Drivers 작성 |
| 2  (Pre Final) | 2024/05/13 | Architectural Drivers 수정  Architecture Design & Evaluation 작성  Architecture Design Description 작성 |
| 3  (Final) | 2024/05/21 | Component Design Description 작성  Architecture Traceability Summary 작성  Appendix 작성 |

**Index**

[1. Project Overview 5](#_heading=h.gjdgxs)

[1.1. Project Background 5](#_heading=h.30j0zll)

[1.2. Business Context Diagram 5](#_heading=h.1fob9te)

[1.3. Stakeholder List 7](#_heading=h.3znysh7)

[1.4. Business Goal List 7](#_heading=h.2et92p0)

[2. System Overview 10](#_heading=h.tyjcwt)

[2.1. System Context Diagram 10](#_heading=h.3dy6vkm)

[2.2. External Entity List 10](#_heading=h.1t3h5sf)

[2.3. External Interface List 13](#_heading=h.4d34og8)

[2.4. System Feature List 18](#_heading=h.2s8eyo1)

[2.5. Domain Model 20](#_heading=h.17dp8vu)

[2.6. Assumptions about the System 21](#_heading=h.3rdcrjn)

[3. Architectural Drivers 23](#_heading=h.26in1rg)

[3.1. Primary Functionality 23](#_heading=h.lnxbz9)

[3.1.1. Use Case Diagram 23](#_heading=h.1bu9ngypd6fk)

[3.1.2. Actor List 23](#_heading=h.3f9op4gf1dm)

[3.1.3. Use Case List 24](#_heading=h.44sinio)

[3.1.4. UC-01 스마트폰 보안 기능 동작 27](#_heading=h.2jxsxqh)

[3.1.5. UC-02 스마트폰 재판 자료 조회 29](#_heading=h.z337ya)

[3.1.6. UC-03 법원TV로 영상 전송 32](#_heading=h.3j2qqm3)

[3.1.7. UC-04 영상 법정 연동 33](#_heading=h.1y810tw)

[3.1.8. UC-05 자료 승인 요청 35](#_heading=h.4i7ojhp)

[3.1.9. UC-06 재판부 자료 승인 38](#_heading=h.2xcytpi)

[3.1.10. UC-07 법원 자료 업데이트 39](#_heading=h.1ci93xb)

[3.1.11. UC-08 공용 영상 제어 40](#_heading=h.3whwml4)

[3.1.12. UC-09 시스템 연동 모니터링 42](#_heading=h.2bn6wsx)

[3.1.13. UC-10 재판 정보 설정 43](#_heading=h.qsh70q)

[3.2. Quality Attribute Scenario 45](#_heading=h.3as4poj)

[3.2.1. The QAS List 46](#_heading=h.1pxezwc)

[3.2.2. QAS-01 영상재판 Latency 최소화 46](#_heading=h.49x2ik5)

[3.2.3. QAS-02 방청객 모바일 앱 사용자의 응답 속도 보장 48](#_heading=h.2p2csry)

[3.2.4. QAS-03 각종 기능 추가 및 확장시 문제 없음 보장 49](#_heading=h.147n2zr)

[3.2.5. QAS-04 오류 없는 원활한 재판 진행 50](#_heading=h.3o7alnk)

[3.2.6. QAS-05 시스템 장애 탐지 및 복구 51](#_heading=h.23ckvvd)

[3.2.7. QAS-06 편리한 재판 자료 제어 52](#_heading=h.ihv636)

[3.2.8. QAS-07 재판 자료 접근 제어 53](#_heading=h.32hioqz)

[3.3. Constraint 54](#_heading=h.vx1227)

[3.3.1. Business Constraint List 55](#_heading=h.3fwokq0)

[3.3.2. Technical Constraint List 55](#_heading=h.1v1yuxt)

[4. Architecture Design & Evaluation 57](#_heading=h.4f1mdlm)

[4.1. Candidate Designs per QA 57](#_heading=h.2u6wntf)

[4.1.1. Candidate Design List 57](#_heading=h.19c6y18)

[4.1.2. QA1: Performance 57](#_heading=h.3tbugp1)

[4.1.3. QA2: Extensibility 65](#_heading=h.55glkc3wyyp4)

[4.1.4. QA3: Availability 71](#_heading=h.nmf14n)

[4.1.5. QA4: Usability 77](#_heading=h.37m2jsg)

[4.1.6. QA5: Security 83](#_heading=h.1mrcu09)

[4.2. Candidate Designs Evaluation for all QAs 88](#_heading=h.2lwamvv)

[4.3. Design Decision 90](#_heading=h.gy19wzn4fh0t)

[5. Architecture Design Description 92](#_heading=h.111kx3o)

[5.1. Architecture Overview 92](#_heading=h.3l18frh)

[5.1.1. Architecture Overview Diagram 92](#_heading=h.206ipza)

[5.1.2. Node Specification 92](#_heading=h.4k668n3)

[5.1.3. Execution Environment Specification 94](#_heading=h.2zbgiuw)

[5.1.4. Communication Path Specification 95](#_heading=h.1egqt2p)

[5.2. Structure View 96](#_heading=h.3ygebqi)

[5.2.1. Static Structure Model 97](#_heading=h.2dlolyb)

[5.2.2. CourtMainView 101](#_heading=h.sqyw64)

[5.2.3. CourtMainController 102](#_heading=h.3cqmetx)

[5.2.4. CourtSecurityController 103](#_heading=h.1rvwp1q)

[5.2.5. CourtDataModel 104](#_heading=h.nb0y97qk7cyi)

[5.2.6. AudienceRequestHandler 105](#_heading=h.i7x5e8wo282g)

[5.2.7. LitigantsRequestHandler 106](#_heading=h.6d4w8q4gdzlh)

[5.2.8. JudgeRequestHandler 107](#_heading=h.3hhem3bkv0hq)

[5.2.9. SystemManagerRequestHandler 108](#_heading=h.kg7fbs34txqg)

[5.2.10. CourtDataManagementService 109](#_heading=h.xizth8ke7cas)

[5.2.11. CourtServiceControlService 111](#_heading=h.2c67y41yzlyb)

[5.2.12. ScreenControlService 112](#_heading=h.mcrbuwwwe3ox)

[5.2.13. MultimediaDeviceControlService 113](#_heading=h.7uvs6f80dl06)

[5.2.14. SecurityManagementService 115](#_heading=h.t3lg1rb6tujh)

[5.2.15. InMemoryCachingService 115](#_heading=h.fjwf1keu2hlx)

[5.2.16. OnlineCourtService 117](#_heading=h.ix4f2rmmv8mz)

[5.2.17. CourtBroadcastingService 118](#_heading=h.xl26v122u3bk)

[5.2.18. CourtDispatcherService 119](#_heading=h.5n1e7agw6go)

[5.2.19. CourtInfoManagementService 121](#_heading=h.yea5yfz00k32)

[5.2.21. CourtDataHandler 122](#_heading=h.2up8jo8dx9up)

[5.2.22. CourtHealthMonitoringService 123](#_heading=h.tulenozliak)

[5.2.23. SystemLogDBHandler 124](#_heading=h.jfcd00vr60hl)

[5.3. Behavior View 126](#_heading=h.4bvk7pj)

[5.3.1. UC-02 스마트폰 재판 자료 조회 126](#_heading=h.2r0uhxc)

[5.3.2. UC-04 영상 법정 연동 127](#_heading=h.1664s55)

[5.3.5. UC-05 자료 승인 요청 127](#_heading=h.3q5sasy)

[5.3.8. UC-06 재판부 자료 승인 127](#_heading=h.25b2l0r)

[5.3.11. UC-07 법원 자료 업데이트 127](#_heading=h.kgcv8k)

[5.4. Deployment View 127](#_heading=h.34g0dwd)

[5.4.1. Artifact Deployment Model 128](#_heading=h.1jlao46)

[5.4.2. Artifact Definition Model 129](#_heading=h.2iq8gzs)

[6. Component Design Description 132](#_heading=h.xvir7l)

[6.1. Component 1 Design Description 132](#_heading=h.3hv69ve)

[6.1.1. Overview 132](#_heading=h.1x0gk37)

[6.1.2. Component Structure Diagram 133](#_heading=h.4h042r0)

[6.1.3. Element List 134](#_heading=h.2w5ecyt)

[6.1.4. Design Rationale 134](#_heading=h.1baon6m)

[6.1.5. Component Behavior Diagram 135](#_heading=h.3vac5uf)

[6.2. Component 2 Design Description 136](#_heading=h.2afmg28)

[6.3. Component 3 Design Description 136](#_heading=h.pkwqa1)

[7. Architecture Traceability Summary 137](#_heading=h.39kk8xu)

[7.1. Architecture Traceability Graph 137](#_heading=h.1opuj5n)

[7.2. Summary of Traceability Items 138](#_heading=h.48pi1tg)

# Project Overview

## Project Background

기존 법정 시스템의 아날로그 방식은 문서와 데이터의 저장, 보관, 검색과 확인 등의 어려움이 있었고, 이로 인해 재판 시간이 늘어나는 등의 문제가 있었습니다. 또한 방청객들도 재판에서 활용되는 내용을 제대로 보기 어렵거나, 영상 재판을 활용하는데도 한계가 있었습니다.



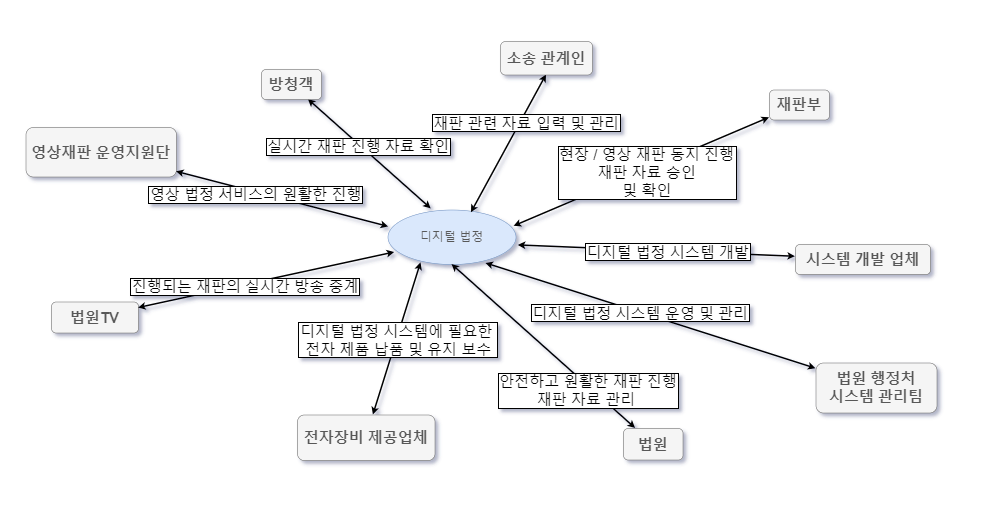
이러한 사회적 배경을 바탕으로 디지털 법정 시스템을 도입하여 문제를 해결하고자 합니다.

본 시스템의 핵심 기능은 다음과 같습니다.

* 재판 관련자는 현장/영상 재판을 동시 진행 할 수 있습니다.
* 소송관계인은 재판과 관련된 멀티미디어 자료를 관련자 PC를 통해 확인, 입력, 관리 및 제어가 가능합니다.
* 방청객은 재판과 관련된 멀티미디어 자료를 법정 내 각종 스크린 뿐만 아니라 모바일 앱으로 실시간으로 확인 가능합니다.
* 재판을 시청 하고자 하는 사람은 상황에 따라 지정된 장소 및 TV로 실시간으로 재판 시청이 가능 합니다.

디지털 법정 시스템의 개발로 공판중심주의를 크게 강화할 수 있게 되었고, 최첨단 IT기술을 이용하여 재판업무의 신속성과 효율성도 향상될 것으로 기대 됩니다.

## Business Context Diagram



## Stakeholder List

|  |  |
| --- | --- |
| **Stakeholder** | **Description** |
| 방청객 | 설명: 법정에서 재판을 방청하고, 관련된 정보를 쉽게 확인할 수 있길 바라는 사용자  관심사: 관심있는 재판에 참석하며, 본인 모바일 장치를 통해 재판 자료 중 관련 자료를 확인 할 수 있길 원함 |
| 소송 관계인 | 설명: 본 시스템을 사용하는 재판 관련 이해 관계자 및 증인  관심사: 법정 진행시 본인이 필요한 데이터를 입력/확인/관리하고 이를 통해 재판이 자신의 목표에 맞게 결정 되길 원함 |
| 재판부 | 설명: 본 시스템을 사용하는 합의부 재판장 혹은 단독판사  관심사: 현장/영상 재판을 효율적이고 원활하게 진행하길 원함 |
| 시스템 개발 업체 | 설명: 본 시스템을 개발, 구축, 유지 보수를 담당  관심사: 사용자(방청객, 소송 관계인, 판사, 법원 등)에게 안정적이고 사용자 친화적인 시스템을 제공하길 원하며, 향 후 확장성 있는 시스템을 개발 하여 오류 없는 원활한 재판 진행을 할 수 있게 하길 원함 |
| 법원 행정처 시스템 관리팀 | 설명: 본 시스템을 관리, 운영을 담당  관심사: 본 시스템을 통해 편리 하게 법정이 운영될 수 있기를 바라며, 안정적으로 시스템이 운영되기를 바람 |
| 법원 | 설명: 효율적으로 법정 운영을 진행 하려는 법원  관심사: 본 시스템을 이용하여, 법원 운용의 효율성을 높여, 사법부의 신뢰와 국민 알 권리 보장 및 국민 사법 접근성 향상을 원함 |
| 법원TV | 설명: 법원 내에서 재판의 온라인 생중계 서비스를 담당하는 관리 조직  관심사: 본 시스템과의 지속적인 연동을 통해 안정적인 온라인 생중계 서비스의 제공 및 서비스 확산을 원함 |
| 영상 재판 운영 지원단 | 설명: 법원 내에서 영상 법정 서비스를 담당하는 관리 조직  관심사: 본 시스템과의 지속적인 연동을 통해 안정적인 영상 법정 서비스의 제공 및 서비스 확산을 원함 |
| 전자장비 제공업체 | 설명: 본 시스템에 사용되는 전자 장비를 납품 및 유지보수 하는 업체  관심사: 본 시스템에서 사용되는 전자 장비를 안정적으로 납품하여 매출 성장을 원함 |

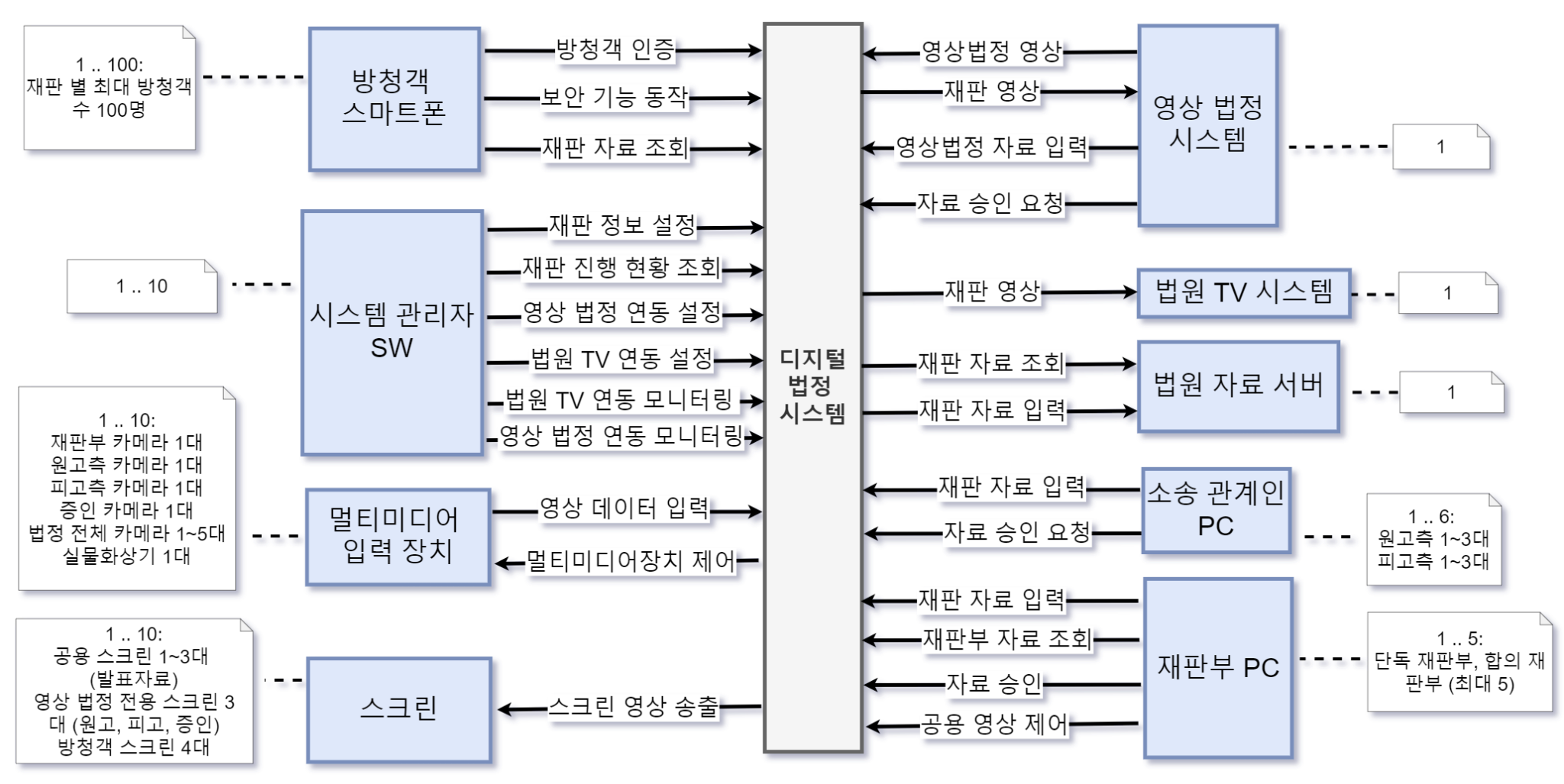
## Business Goal List

|  |  |  |  |
| --- | --- | --- | --- |
| **Stakeholder** | **Business Goal** | | |
| **ID** | **Statement** | **I** |
| 방청객 | BG-01 | 본 시스템을 통해 필요한 재판 정보를 쉽고 빠르게 확인 할 수 있길 원함   * 방청객은 디지털 법정 전용 WIFI 연결을 통해 휴대폰 요금제와 상관없이 필요한 기능을 사용하길 원함 * 방청객은 WIFI 연결 후 사용자 앱의 설치 및 사용 방법을 볼 수 있길 원함 * 방청객은 앱 설치 후 앱을 통해 관련 재판 정보를 신속하게 확인 할 수 있길 원함 | 중 |
| 소송 관계인 | BG-02 | 재판 진행 시, 필요한 정보를 문제 없이 입력/확인 관리 할 수 있길 원함   * 재판 중 필요한 자료 입력이 가능하길 원함 * 입력한 자료가 재판에 사용될 때, 조작 및 딜레이 없이 활용 될 수 있길 원함 | 상 |
| 재판부 | BG-03 | 본 시스템을 통해 법정 진행 데이터를 입력/확인/관리 하여 효율적인 재판 진행을 하길 원함   * 판례 확인 및 입력 자료 승인 기능이 가능 하길 원함 * 기존 대비 재판 진행 시간 10% 단축 되길 원함 * 기존 대비 재판 기간 10% 단축 되길 원함 | 상 |
| 시스템 개발 업체 | BG-04 | 본 시스템과 연동될 시스템(법원 자료 서버, 온라인 생중계 서비스, 영상 법정 서비스)들이 안정적으로 운용되길 원하며, 각종 기능 추가 및 확장, 시스템 동시 사용자 수 증가에 따른 비용 최소화를 기대 함   * 법원 자료 서버, 온라인 생중계 서비스, 영상 법정 서비스 간의 안정적인 상호 운용성을 원함 * 각종 스크린/모니터/멀티미디어 장치, 관련자 PC 등이 변경/확장 되더라도 시스템 운용이 원활 하길 원함 * 디지털 재판 및 사용자 수 증가에 따른 비용 최소화 | 중 |
| 법원 행정처  시스템 관리팀 | BG-05 | 본 시스템을 통해 디지털 재판 설정을 할 수 있길 원하며 설정된 행태로 운영되기를 기대한다 또한 문제 발생시 데이터 손실 없이 시스템이 복구 되길 원함   * 재판의 형태(현장/영상) 및 재판 생중계 여부 설정을 할 수 있길 원함 * 방청객이 올바른 재판 자료를 조회 하여 볼 수 있길 원함 * 재판 시작 부터 종료 시점 사이에 시스템 가동 시간이 99.9% 이상이길 원함 * 시스템 장애 발생시 데이터 손실 없이 복구 되길 원함 | 상 |
| 법원 | BG-06 | 본 시스템을 이용하여 안정적으로 재판 진행이 가능 하길 원함   * 재판 중 입력/변경한 정보가 안정적으로 관리되길 원함   본 시스템을 이용하여 재판 정보가 임의로 외부 노출 되는 것이 방지될 수 있길 원함   * 재판 중 불필요한 방해가 발생되지 않길 원함 * 재판 중인 정보가 외부로 유출되지 않길 원함 | 상 |
| 법원TV | BG-07 | 본 시스템과의 연동을 통해, 온라인 생중계 서비스의 확산 및 시청자 만족도 향상을 원함   * 재판부, 소송관계인, 발표자료 화면을 각각 다른 소스로 받아 시청자 친화적인 방송을 할수 있길 원함 | 중 |
| 영상재판 운영 지원단 | BG-08 | 본 시스템과의 안정적인 연동을 통해, 영상재판 서비스의 확산 및 사용자 만족도 향상을 원함   * 디지털 법정 시스템이 제공하는 기능의 100%가 영상재판에서도 사용될 수 있길 원함 | 중 |
| 전자 장비 제공 업체 | BG-09 | 본 시스템에서 사용되는 모든 전자 장비를 납품하여 매출 성장하고자 하며, 향후 VR 기기 납품을 통해 VR 시장 선점을 원함   * 본 시스템에서 사용되는 전자 장비 납품을 통해 연 매출 10% 향상을 원함 | 하 |

\* I : Importance << 상 중 하로 구분 >>

# System Overview

## System Context Diagram



## External Entity List

|  |  |
| --- | --- |
| **Name** | **Description** |
| 방청객 스마트폰 | ㅁ 유형: Device  ㅁ 역할: 디지털 법정 시스템 내부망 연결, 방청객 스마트폰 앱 다운로드, 설치, 실행 후 아래의 사용자 역할을 수행할 수있게 해 주는 장치  ㅁ 관련 Stakeholder: 방청객  ㅁ 시스템 사양: 네트워크 통신이 가능한 Android Device  ㅁ 방청객 역할: 스마트폰을 활용하여 디지털 법정 시스템 내부망 연결을 하고 앱 다운로드/설치/실행 하여 방청객 인증 후 아래의 기능을 사용 한다.   * 보안 기능 활성화 * 재판 자료 조회   ㅁ 고려사항: 방청객은 방청 승인 일련번호를 전자 법정 모바일 앱에 입력하여 인증을 한 후 재판 관련 자료 조회를 할 수 있다. |
| 시스템 관리자 SW | ㅁ 유형: 시스템  ㅁ 역할: 본 시스템에 연동 되어 다음과 같은 기능을 제공하는 웹 어플리케이션   * 재판 정보 설정 * 재판 진행 현황 조회 * 영상 법정 연동 설정 및 동작 모니터링 * 법원 TV 연동 설정 및 동작 모니터링   ㅁ 관련 Stakeholder: 법원 행정처 시스템 관리팀  ㅁ 시스템 사양: 네트워크 통신이 가능한 데스크탑 PC, Windows 10 탑제  ㅁ 시스템 품질 수준:   * 가용성 99% |
| 멀티미디어 입력장치 | ㅁ 유형: Device  ㅁ 역할: 본 시스템의 제어를 받아 멀티미디어 영상 소스를 입력 시키는 장치   * 카메라: 촬영된 영상을 본 시스템으로 전송 * 실물 화상기: 실물 입체 영상을 본 시스템을 통해 공용 스크린으로 전달   ㅁ 관련 Stakeholder: 법원, 전자장비 제공 업체  ㅁ 시스템 사양:   * 카메라: FHD 해상도 60 fps * 실물 화상기: 500만 화소, 16배 줌   ㅁ 시스템 품질 수준:   * 가용성 99% |
| 스크린 | ㅁ 유형: Device  ㅁ 역할: 본 시스템의 제어를 받아 승인 받은 멀티미디어 자료를 보여주는 장치   * 공용 스크린: 재판부에서 승인한 자료를 화면에 보여줌 * 영상 법정 전용 스크린: 영상법정 시스템에서 들어오는 영상을 보여주는 장치   ㅁ 관련 Stakeholder: 법원, 전자장비 제공 업체  ㅁ 시스템 사양   * 빔 프로젝터: 해상도 4K UHD (3840x2160) , 최대 화면 140인치, 밝기 2200 안시 이상 * TV: 해상도 4K UHD (3840x2160) , 70 인치   ㅁ 시스템 품질 수준:   * 가용성 99% |
| 영상법정시스템 | ㅁ 유형: 시스템  ㅁ 역할: 본 시스템과 연동 하여 다음과 같은 기능을 제공하는 영상 법정 서비스를 운영하는 시스템   * 영상법정의 영상을 본 시스템에 제공 * 본 시스템으로 부터 재판 영상을 받아 영상 법정 진행 * 본 시스템에 재판 자료 입력 * 본 시스템에 재판 자료 승인 요청   ㅁ 관련 Stakeholder: 영상재판 운영지원단  ㅁ 시스템 사양   * HTTP 서비스가 가능한 클라우드 서버   ㅁ 시스템 품질 수준:   * 가용성 99.9% |
| 법원TV 시스템 | ㅁ 유형: 시스템  ㅁ 역할: 본 시스템과 연동하여 재판 영상을 확보하고 이를 TV로 중계 해 주는 시스템  ㅁ 관련 Stakeholder: 법원TV  ㅁ 시스템 사양   * HTTP 서비스가 가능한 클라우드 서버   ㅁ 시스템 품질 수준:   * 가용성 99% * 20초 이내로 법원 영상이 법원 TV 시스템에 전달 되어야 함 * 전달 받은 영상은 10초 이내로 TV로 중계 되어야 함 |
| 법원 자료 서버 | ㅁ 유형: 시스템  ㅁ 역할: 본 시스템의 자료 조회/입력 요청을 받아 결과를 전달하는 시스템  ㅁ 관련 Stakeholder: 법원  ㅁ 시스템 사양   * HTTP 서비스가 가능한 클라우드 서버   ㅁ 시스템 품질 수준:   * 가용성 99.9% * 재판 자료 조회 요청 시 1초 이내로 결과 반환 (멀티미디어 자료 및 용량이 큰 자료의 경우 네트워크 상태에 따라 달라질 수 있음) * 재판 자료 입력시 1초 이내로 결과 반환 (멀티미디어 자료 및 용량이 큰 자료의 경우 네트워크 상태에 따라 달라질 수 있음) |
| 소송 관계인 PC | ㅁ 유형: Device  ㅁ 역할: 소송 관계인이 web browser를 통해 본 시스템에 접속 후, 재판 자료 입력 및 승인 요청을 할 수 있는 장치  ㅁ 관련 Stakeholder: 소송 관계인  ㅁ 시스템 사양: 네트워크 통신이 가능한 데스크탑 PC, Windows 10 탑제  ㅁ 시스템 품질 수준:   * 가용성 99% |
| 재판부 PC | ㅁ 유형: Device  ㅁ 역할: 재판부가 web browser를 통해 본 시스템에 접속 후, 재판 자료 입력/승인 및 재판부 자료 조회, 공용 스크린 영상 조작을 할 수 있는 장치  ㅁ 관련 Stakeholder: 재판부  ㅁ 시스템 사양: 네트워크 통신이 가능한 데스크탑 PC, Windows 10 탑제  ㅁ 시스템 품질 수준:   * 가용성 99% |

## External Interface List

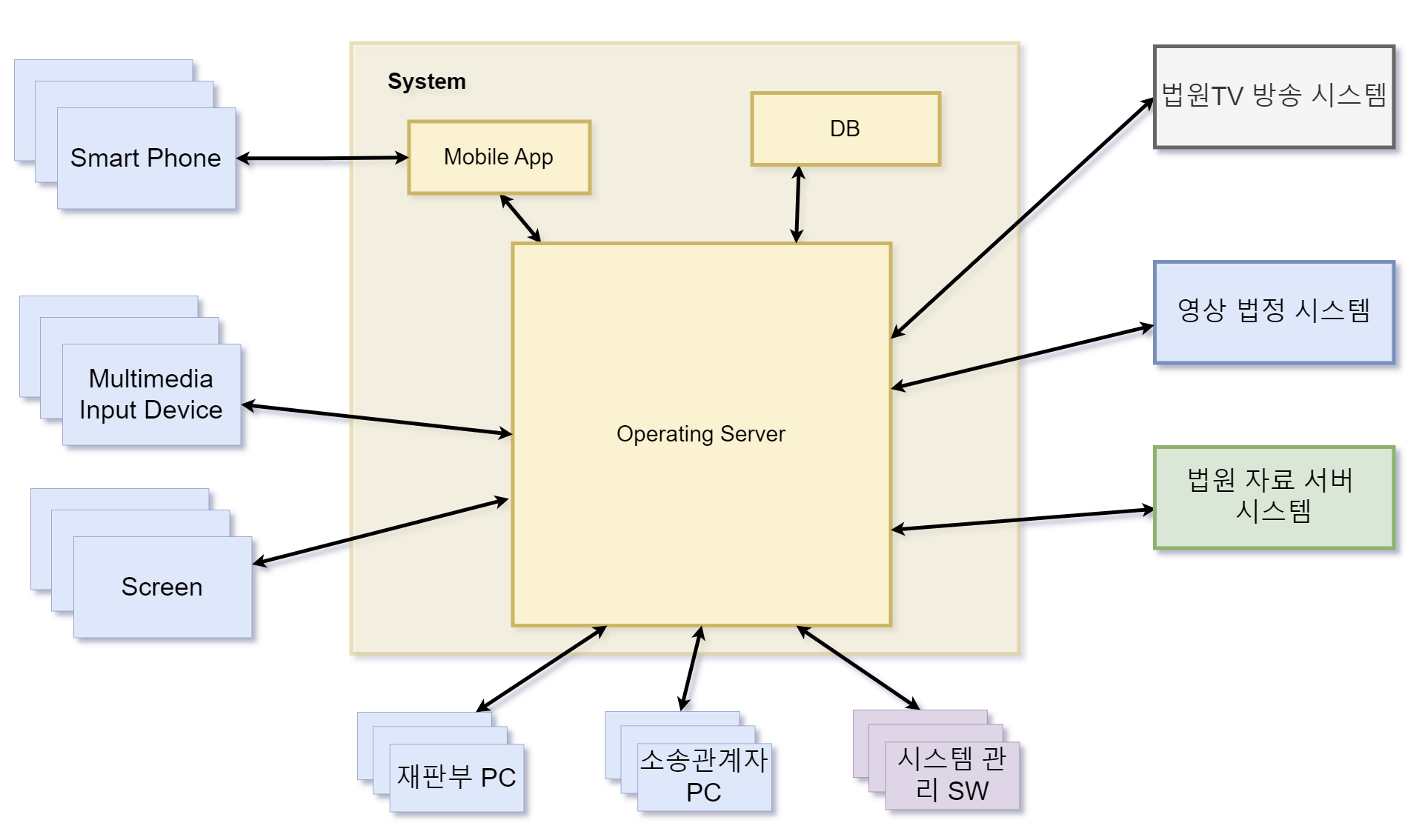
|  |  |
| --- | --- |
| **Name** | **Description** |
| 방청객 인증 | 역할: 스마트폰을 통해 방청객 일련번호를 받아, 모바일 앱을 통해 등록된 방청객인지 확인 한다.  User Interface : Mobile Application UI  Device Interface: 터치 스크린 또는 키패드  특성   * 일련번호 길이보다 길면 더이상 입력 되지 않는다. * 일련번호(숫자/알파벳) 포맷과 맞지 않으면 입력 되지 않는다. |
| 보안 기능 동작 | 역할: 스마트폰을 통해 보안 기능 동작 수행 요청을 받아 모바일 앱을 통해 아래의 보안 기능을 수행한다.   * 어플리케이션 보안 기능 수행 권한 요청 * 카메라 기능 잠금 * 화면 녹화 및 캡쳐 기능 잠금 * 소리 모드 무음으로 변경   User Interface : Mobile Application UI  Device Interface: 터치 스크린 또는 키패드  특성   * 보안 기능 동작 수행 시 어플리케이션이 권한이 없는 경우 사용자가 권한을 줄수 있다. |
| 재판 자료 조회 | 역할: 스마트폰을 통해 승인된 재판 관련 자료 리스트를 요청 받을 시, 모바일 앱을 통해 사용자에게 보여준다.  User Interface : Mobile Application UI  Device Interface: 터치 스크린 또는 키패드  System Interface : HTTPS  특성   * 자료 리스트는 파일 이름 및 링크로 보여진다. * 사용자가 선택시 연결된 Viewer 앱으로 실행 된다. |
| 재판 정보 설정 | 역할: 시스템 관리자 SW가 본 시스템을 통해 재판 정보를 설정한다. 재판 정보의 구성은 아래와 같다.   * 재판 시간 * 재판부 정보 * 소송 관계인 정보 * 재판 자료 * 영상재판연동 정보 * 법원TV연동 정보   User Interface : Consol (Web Application)  System Interface : HTTPS  특성   * 데이터 포맷: JSON * 데이터 크기: 1MB |
| 재판 진행 현황 조회 | 역할: 시스템 관리자 SW가 본 시스템을 통해 등록된 재판 정보 및 현재 상태를 조회 한다. 조회 가능한 정보는 아래와 같다.   * 재판 시간 * 재판부 정보 * 소송 관계인 정보 * 재판 자료 * 영상재판연동 정보 * 법원TV연동 정보   User Interface : Consol (Web Application)  System Interface : HTTPS  특성   * 데이터 포맷: JSON * 데이터 크기: 1MB |
| 영상 법정 연동 설정 | 역할: 시스템 관리자 SW가 본 시스템에 등록된 재판을 영상 법정으로 진행되게 할 것인지 설정한다.  User Interface : Consol (Web Application)  System Interface : HTTPS  특성   * 영상 재판으로 등록된 재판을 진행하고자 하는 경우, 재판 정보를 영상 법정 시스템에 전달하고 영상 법정 ID를 발급 받아 맵핑 시킨다. * 데이터 포맷: JSON * 데이터 크기: 1MB |
| 법원 TV 연동 설정 | 역할 시스템 관리자 SW가 본 시스템에 등록된 재판을 법원TV로 재판 진행 영상을 송출 할 것인지 설정한다.  User Interface : Consol (Web Application)  System Interface : HTTPS  특성   * 재판이 방송 중계가 될 수 있게 진행하고자 하는 경우, 재판 정보를 법원TV 시스템에 전달하고 방송 중계 ID를 발급 받아 맵핑 시킨다. * 데이터 포맷: JSON * 데이터 크기: 1MB |
| 영상 데이터 입력 | 역할: 멀티미디어 입력 장치로 부터 촬영된 레코딩 영상을 전달 받는다.  Device Interface : 카메라/실물 화상기  System Interface : RTP/UDP  특성   * 크기: 수십 MB (영상 데이터) * 영상 품질: UHD 32Mbps(60fps) * 가용성 99.9% |
| 멀티미디어장치 제어 | 역할: 멀티미디어 입력 장치의 카메라 레코딩 제어 명령을 전달 한다.  Device Interface : 카메라/실물 화상기  System Interface : RTP/UDP  특성   * 크기: 1MB (장치 제어 명령) * 가용성 99.9% |
| 영상 법정 영상 | 역할: 본 시스템으로 영상 법정의 영상 데이터를 받는다.  System Interface : RTP/UDP  특성   * 크기: 수십 MB (영상 데이터) * 영상 품질: UHD 32Mbps(60fps) * 가용성 99.9% |
| 재판 영상 | 역할: 재판 영상이 필요한 시스템(영상 법정 시스템/법원TV 시스템)이 본 시스템으로 재판 영상 자료를 요청 한다. 요청시 아래의 정보를 포함한다.   * 영상 정보 * 영상 Data 전달 API   System Interface : HTTPS  특성:   * 데이터 포맷: JSON * 데이터 크기: 1MB |
| 영상 법정 자료 입력 | 역할: 영상 법정 시스템으로 부터 재판 관련 file을 입력 받는다.  System Interface : HTTPS  특성   * 첨부 파일 용량: 개당 최대 100MB로 제한 (time out: 30 sec) * 전송 완료 시 callback 호출 |
| 자료 승인 요청 | 역할: 본 시스템을 통해 소송 관계인이 입력된 재판 자료가 재판에 사용될 수 있도록 재판부로 자료 승인을 요청한다. 자료 승인 요청은 영상 법정을 통해서 올 수도 있고, 소송 관계인 PC에서 올 수도 있다.  User Interface : Consol (Web Application)  System Interface : HTTPS  특성   * 데이터 포맷: JSON * 데이터 크기: 1MB |
| 재판 자료 입력 | 역할: 소송 관계인/재판부 PC로 부터 재판 관련 file을 입력 받는다.  User Interface : Consol (Web Application)  특성   * 첨부 파일 용량: 개당 최대 100MB로 제한 (time out: 30 sec) * 전송 완료 시 callback 호출 |
| 재판부 자료 조회 | 역할: 재판부 PC에서는 본 시스템을 통해 현재 재판 관련 자료를 조회 하여 볼 수 있다.  User Interface : Consol (Web Application)  System Interface : HTTPS  특성   * 재판부 자료 조회의 결과는 승인된 자료와 승인 되지 않은 자료를 모두 포함한다. * 자료 리스트는 파일 이름 및 링크로 보여진다. * 사용자가 선택시 연결된 Viewer 앱으로 실행 된다 * 데이터 포맷: JSON * 데이터 크기: 1MB |
| 자료 승인 | 역할: 재판부 PC에서는 본 시스템을 통해 조회 된 자료의 승인 여부를 결정 한다.  User Interface : Consol (Web Application)  System Interface : HTTPS  특성   * 승인 되지 않은 자료를 승인 하거나 승인된 자료를 승인 취소 한다. * 데이터 포맷: JSON * 데이터 크기: 1MB |
| 공용 영상 제어 | 역할: 재판부 PC에서는 본 시스템을 통해 조회 된 자료중 하나를 선택하여 공용 스크린 화면에서 볼 수 있게 제어 한다.  User Interface : Consol (Web Application)  Device Interface: HDMI  특성   * 재판부 PC 에서 필요한 자료를 선택하여 Viewer를 실행 한다. * Viewer의 실행 화면을 공용 스크린에 보이게 한다. |
| 법원TV 연동 모니터링 | 역할: 시스템 관리자 SW는 본 시스템에 현재 진행 중인 재판의 영상 데이터가 법원TV 로 전달 되는 상태를 요청하여 받는다. 해당 정보는 아래를 포함 한다.   * 비정상 구간 정보 * 각 멀티미디어 입력 장치로 부터 들어오는 영상 및 품질 상태 정보 * 본 시스템에서 법원 TV 시스템 API로 전달 하기 직전의 영상 및 품질 상태 정보   User Interface : Consol (Web Application)  System Interface : RTP/UDP, HTTPS  특성   * 크기: 수십 MB (영상 데이터 + 영상 품질 상태 정보) * 영상 품질: UHD 32Mbps(60fps) * 영상 품질 상태 정보   + Resolution / Bitrate / Frame Rate * 가용성 99.9% |
| 영상 법정 연동 모니터링 | 역할: 시스템 관리자 SW는 본 시스템에 현재 진행 중인 영상 재판의 영상 데이터의 흐름 상태를 요청하여 받는다. 해당 정보는 아래를 포함 한다.   * 비정상 구간 정보 * 각 멀티미디어 입력 장치로 부터 들어오는 영상 및 영상 품질 상태 * 영상 법정 시스템 API로 전달 되기 직전의 영상 및 영상 품질 상태 * 영상 법정 시스템에서 들어오는 영상 및 영상 품질 상태   User Interface : Consol (Web Application)  System Interface : RTP/UDP, HTTPS  특성   * 크기: 수십 MB (영상 데이터 + 영상 품질 상태 정보) * 영상 품질: UHD 32Mbps(60fps) * 영상 품질 상태 정보   + Resolution / Bitrate / Frame Rate * 가용성 99.9% |

## System Feature List

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Title** | **Description** | **I** | **Related**  **Business Goal ID** |
| SF-01 | 스마트폰 방청객 인증 | 방청객 일련번호와 재판정보 매칭 상황을 조회 하여 방청객 인증 하는 모바일 앱 기능 | 중 | BG-01  BG-06 |
| SF-02 | 스마트폰 보안 기능 동작 | 카메라 기능 잠금, 화면 녹화 및 캡쳐 기능 잠금, 소리 모음 모드로 변경 하는 모바일 앱 기능 | 상 | BG-01  BG-06 |
| SF-03 | 스마트폰 재판 자료 조회 | 재판부로 부터 승인 받은 재판 자료 리스트를 조회하여 보여주는 모바일 앱 기능 | 상 | BG-01  BG-06 |
| SF-04 | 재판 정보 설정 | 재판의 기준 정보를 설정하는 기능 | 상 | BG-03  BG-05 |
| SF-05 | 재판 진행 현황 조회 | 재판의 기준 정보를 조회하는 기능 | 중 | BG-03  BG-05  BG-07  BG-08 |
| SF-06 | 법원TV로 영상 전송 | 온라인 생방송을 위해 영상 정보 전달 기능 | 상 | BG-03  BG-05  BG-07 |
| SF-07 | 장치 영상 데이터 수신 | 멀티미디어 입력 장치로 부터 영상 데이터 수신 하여 저장하는 기능 | 상 | BG-06  BG-07  BG-09 |
| SF-08 | 영상법정 영상 데이터 수신 | 영상 법정으로 부터 영상 데이터 수신 하여 이를 현장 법정에 보여주는 기능 | 상 | BG-03  BG-08 |
| SF-09 | 영상법정으로 영상 전송 | 장치 영상을 영상 법정으로 전달 하는 기능 | 상 | BG-03  BG-08 |
| SF-10 | 자료 승인 요청 | 소송관계인 으로부터 재판 자료를 입력하고 저장된 재판 자료를 재판부에 승인 요청하는 기능 | 중 | BG-02  BG-06  BG-08 |
| SF-11 | 법원 자료 업데이트 | 재판 정보를 법원 자료 서버로 부터 조회 하여 져와서 디지털 기준 정보를 맞추고, 재판이 종료되면 변경된 자료를 법원 자료 서버로 업데이트 하는 기능 | 상 | BG-02  BG-03  BG-06 |
| SF-12 | 재판부 자료 승인 | 재판부의 승인 여부와 상관 없이 모든 입력된 자료를 조회 하고 승인 하는 기능 | 중 | BG-03  BG-06 |
| SF-13 | 공용 영상 제어 | 재판 자료 및 현장법정/영상법정 영상 중 하나를 선택하여 공용 스크린으로 재생 화면 송출하는 기능 | 상 | BG-02  BG-03  BG-06  BG-09 |
| SF-14 | 법원TV 연동 모니터링 | 법원TV와의 서비스 연동 상태를 파악하고 문제 발생시 해당 구간을 표시 해주는 기능 | 중 | BG-03  BG-04  BG-05  BG-07 |
| SF-15 | 영상 법정 연동 모니터링 | 영상 법정 진행시, 서비스 연동 상태를 파악하고 문제 발생시 해당 구간을 표시 해주는 기능 | 중 | BG-03  BG-04  BG-05  BG-08 |

\* I: Importance << 상 중 하로 구분 >>

## Domain Model



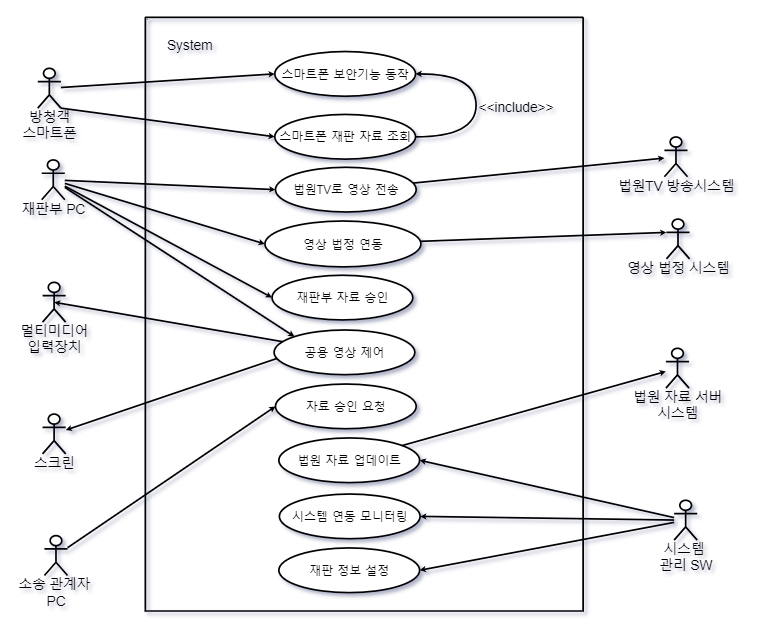
## Assumptions about the System

1. 방청객은 재판 입장 전에 법원 사이트를 통해 방청권 신청을 할 수있고, 승인시 일련번호를 부여 받는다. 이 일련 번호는 방청객 인증시 사용된다.
2. 방청객이 디지털 법정에서 사용하는 앱은 안드로이드 앱스토어를 통해 다운로드 받을 수 있으며, 법원 사이트 및 법정 입구에서 관련 가이드를 확인 할 수 있다.
3. 방청권을 승인 받은 방청객은 법정 입구에서, 스마트폰으로 디지털 법정 시스템과 연결 가능한 네트워크 망 정보와 패스워드를 제공 받는다.
4. 방청객의 스마트폰은 모두 안드로이드 OS를 사용하는 것으로 가정한다.
5. 재판부 중 1인은 중급 이상의 디지털 법정 시스템 사용자로, 현장에서 재판에 필요한 조작을 진행 한다.

# Architectural Drivers

## Primary Functionality

### Use Case Diagram



### Actor List

|  |  |
| --- | --- |
| Name | Description |
| 방청객 스마트폰 | 방청객이 사용하는 스마트폰으로 디지털 법정 네트워크에 접속하고, 인증 후 재판 자료 조회 한다. 자료 조회를 위해서는 반드시 보안 기능을 활성화 시켜야만 사용 가능하다. |
| 소송 관계자 PC | 소송 관계자가 해당 장치에서 시스템에 접속하여 재판에 필요한 자료를 입력/관리/승인 요청 작업을 수행한다. |
| 재판부 PC | 재판부가 해당 장치에서 시스템에 접속하여 재판에 필요한 자료를 입력/관리/승인 하며, 공용 스크린 제어를 통해 재판 참석자가 모두 자료를 볼 수 있도록 관리 한다. |
| 법원TV 방송 시스템 | 시스템관리자가 연동요청을 해 둔 경우, 법원TV 방송 시스템으로 재판 영상이 송출되고 이를 받아 방송 시스템 내부에서 재판 방송을 진행 한다. |
| 영상 법정 시스템 | 시스템관리자가 재판의 형태를 영상재판으로 설정한 경우, 영상 재판시스템과 연동되어 영상법정으로 재판이 진행되게 한다. 소송 관계자는 영상 법정 시스템을 통해 재판 자료를 입력/관리/승인 요청을 수행 한다. |
| 법원 자료 서버 시스템 | 시스템 관리자가 재판 정보 설정 시, 해당 재판의 이전 자료들을 법원 자료 서버로 부터 조회 하여 내부에 캐싱 해 둔다. 이후 재판이 종료되고 재판부로 부터 승인된 자료들은 법원 자료 입력을 통해 법원 자료 서버로 업데이트 된다. |
| 시스템관리 SW | 시스템 관리자가 사용하는 SW로 여기서는 법원 자료 서버 시스템, 영상 법정 시스템, 법원TV 방송 시스템의 연동 정보를 가져와 디지털 법정의 정보를 입력/조회/관리 할 수 있다. 여기서 저장된 정보를 기반으로 영상/현장 재판의 타입과 방송 중계의 여부가 결정되며, 설정에 따라 재판 시작 시 시스템 연동이 시작된다. |
| 멀티미디어 입력 장치 | 재판부 PC를 통해 제어되는 입력 장치이며, 이는 시스템 관리자가 설정한 기준 정보에 따라 받은 데이터는 법원 TV 방송 시스템이나 영상 법정 시스템으로 전달 된다. |
| 스크린 | 재판부 PC를 통해 제어되는 출력 장치이며, 재판부의 결정에 따라 확정된 데이터를 현장 법정에서 재판 관련자들이 볼 수 있게 스크린으로 보여준다. |

### Use Case List

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Title** | **Summary of Description** | **Priority** | | **System**  **Feature ID** | **ASR?** |
| **I** | **D** |
| UC-01 | 스마트폰 보안기능 동작 | 인증 받은 방청객의 스마트폰에서, 카메라 기능 잠금, 화면 녹화 및 캡쳐 기능 잠금, 소리 모음 모드로 변경 하는 모바일 앱 기능 | 상 | 중 | SF-01  SF-02 | X |
| UC-02 | 스마트폰 재판 자료 조회 | 재판부로 부터 승인 받은 재판 자료 리스트를 조회하여 보여주는 모바일 앱 기능 | 상 | 중 | SF-03 | O |
| UC-03 | 법원TV로 영상 전송 | 온라인 생방송을 위해 영상 정보 전달 기능 | 상 | 중 | SF-06 | X |
| UC-04 | 영상 법정 연동 | 영상 법정 시스템과 연동을 시작하고 각 시스템에 필요한 영상을 주고 받으며 영상 재판을 시작 및 종료 하는 기능 | 상 | 상 | SF-08  SF-09 | O |
| UC-05 | 자료 승인 요청 | 재판 자료를 입력하고 재판부에 승인 요청하는 기능 | 중 | 하 | SF-10 | X |
| UC-06 | 재판부 자료 승인 | 재판과 관련하여 승인 요청된 자료를 조회하고 해당 자료의 승인 여부를 결정하는 기능 | 상 | 중 | SF-12 | X |
| UC-07 | 법원 자료 업데이트 | 디지털 재판 시작 전/후에 법원 자료 서버로 재판 정보를 업데이트 하여 자료 히스토리가 관리될 수 있게 하는 기능 | 상 | 중 | SF-11 | O |
| UC-08 | 공용 영상 제어 | 공유 가능한 법정 데이터를 확인해서, 이를 법정 공용 스크린으로 화면 공유하는 기능 | 상 | 하 | SF-07  SF-13 | X |
| UC-09 | 시스템 연동 모니터링 | 재판의 기준 정보를 확인하고 해당 재판이 법원TV로 생중계 되거나, 영상법정으로 재판이 진행 되는 경우, 시스템의 구간별 기능 동작 여부를 모니터링 하는 기능 | 중 | 상 | SF-05  SF-14  SF-15 | X |
| UC-10 | 재판 정보 설정 | 재판의 기준 정보를 설정하여 해당 재판이 필요에 맞게 동작 되도록 하는 기능 | 상 | 상 | SF-04  SF-05 | X |

\* I: Importance (Business 관점) D: Difficulty (Techniques 관점) << 상 중 하로 구분 >>

### UC-01 스마트폰 보안 기능 동작

#### Scenario List

|  |  |
| --- | --- |
| **Scenario Title** | **Scenario Description** |
| 스마트폰 보안 기능 활성화 | 스마트폰 보안 기능을 활성화 시킨다 |
| 앱 권한 없음 | 스마트폰 앱이 해당 디바이스에 보안 기능을 활성화 할 권한이 없는 경우이다. 사용자에게 권한 승인을 요청 한다. |
| 앱 권한 요청 거절 | 스마트폰 앱의 보안 세팅을 위한 권한 요청이 거절된 경우이다. 사용자의 권한 승인 거절 결과를 보여준다. |

#### Use Case Description

|  |  |
| --- | --- |
| **Use Case** | 스마트폰 보안기능 동작 |
| **Actor** | A. 방청객 스마트폰 |
| **Description** | 방청객은 방청객 스마트폰을 통해 보안 기능 동작 요청을 하고, 앱은 스마트폰 보안 기능을 실행 한다. |
| **Stakeholders** | 방청객, 법원 |
| **Preconditions** | 방청객 스마트폰은 전자 법정 시스템 전용 망에 연결 됨  방청객 스마트폰은 방청객 인증을 완료 함 |
| **Main Scenario** | (A) 방청객 스마트폰, (S) 시스템 (Mobile App)  1. (A)는 방청객은 보안 기능을 실행한다.  2. (S)는 (A)의 카메라 기능을 Disable 시킨다.  3. (S)는 (A)의 화면 녹화 및 캡쳐 기능을 Disable 시킨다.  4. (S)는 (A)의 소리 모음 모드로 변경하고 해당 기능 설정 기능을 Disable 시킨다.  5. (S)는 (A)에게 보안 기능 설정이 완료 되었음을 알리고 본 케이스를 종료한다. |
| **Alternative Scenario** | AF1. 앱 권한 없음  분기점: Main Scenario 2   1. (S)는 (A)에게 카메라/화면녹화 및 캡쳐/ 소리 모드 변경 권한을 요청한다   AF2. 앱 권한 요청 거절  분기점: Main Scenario 3   1. (S)는 (A)에게 보안기능 설정이 실패했음을 보여준 후 본 케이스를 종료한다. |

#### System Sequence Diagram

****

|  |  |
| --- | --- |
| **System Interface / Operation** | **Description** |
| *activateMobileSecurity( )* | 휴대폰 보안 기능 실행 |
| *appPermissionConfirmed()* | 시스템에게 보안 기능 설정에 필요한 권한 확인   * permissionInfo JSON 파일을 파싱하여 승인된 내역 확인후   + 전체 승인이 된 경우 -> 권한 승인   + 하나라도 승인이 안된 경우 -> 권한 요청 거절 |

### UC-02 스마트폰 재판 자료 조회

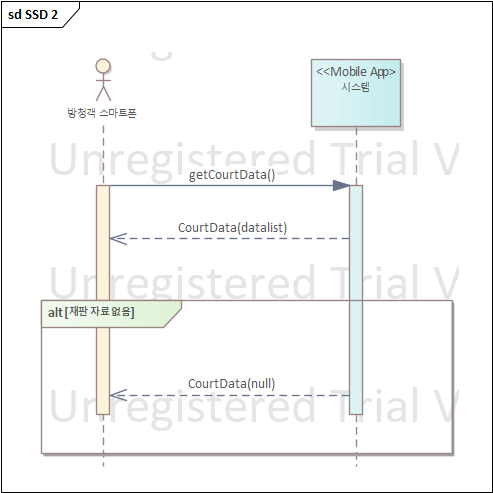
#### Scenario List

|  |  |
| --- | --- |
| **Scenario Title** | **Scenario Description** |
| 스마트폰 재판 자료 조회 | 방청객 스마트폰을 통해 재판 자료 조회를 요청하여 리스트를 받는다. |
| 재판 자료 없음 | 재판 자료 요청 확인시 등록된 재판 자료가 없는 경우 이다. 방청객 스마트폰에 등록된 자료가 없음을 보여준다. |

#### Use Case Description

|  |  |
| --- | --- |
| **Use Case** | 스마트폰 재판 자료 조회 |
| **Actor** | A. 방청객 스마트폰 |
| **Description** | 방청객 스마트폰을 통해 재판 자료 조회를 요청하여 리스트를 받는다. |
| **Stakeholders** | 방청객, 법원 |
| **Preconditions** | 방청객 스마트폰은 전자 법정 시스템 전용 망에 연결 됨  방청객 스마트폰은 방청객 인증을 완료 함  방청객 스마트폰의 스마트폰 보안 기능이 설정 됨 |
| **Main Scenario** | (A) 방청객 스마트폰, (S) 시스템(Mobile App)  1. (A)는 재판의 자료 조회를 선택한다.  2. (S)는 조회된 자료를 (A)로 보여준다. |
| **Alternative Scenario** | AF1. 재판 자료 없음  분기점: Main Scenario 2   1. (S)는 (A)에게 조회할 자료가 없음을 알리고 본 케이스를 종료한다. |

#### System Sequence Diagram

****

|  |  |
| --- | --- |
| **System Interface / Operation** | **Description** |
| *getCourtData( )* | 재판 자료 가져오기 기능 |

### UC-03 법원TV로 영상 전송

#### Scenario List

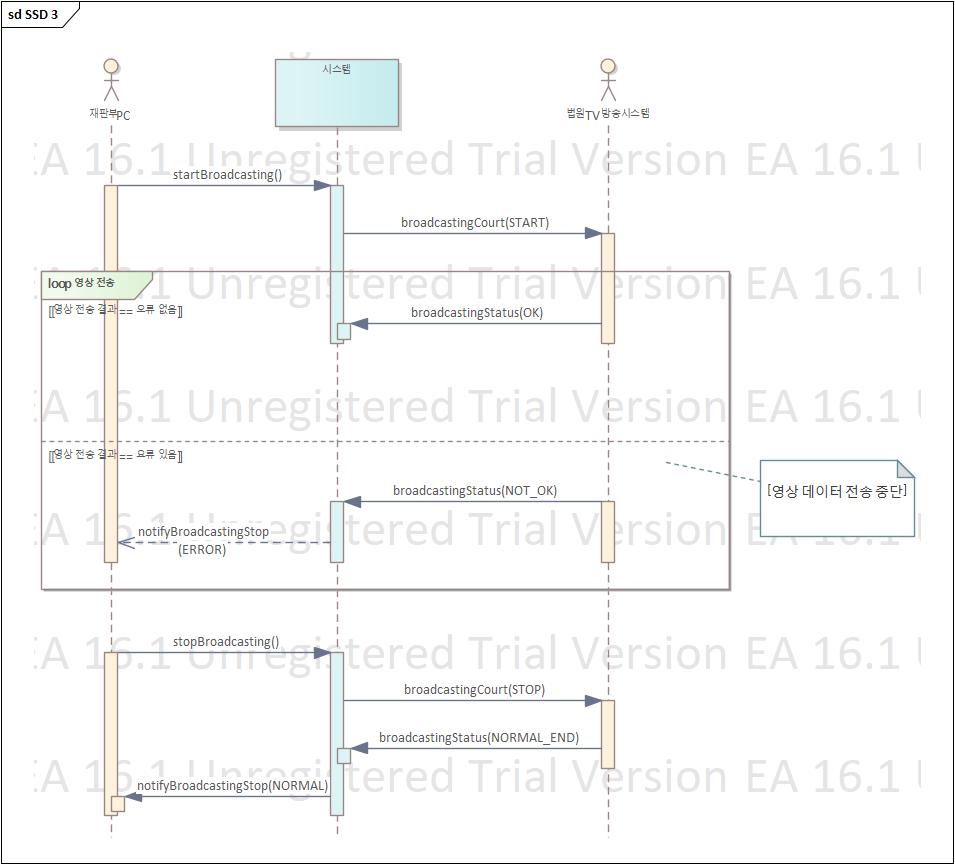
|  |  |
| --- | --- |
| **Scenario Title** | **Scenario Description** |
| 법원TV 영상 전송 | 법원TV 시스템으로 영상 정보를 전달 한다. |
| 영상 데이터 전송 중단 | 법원TV 방송시스템으로 영상 정보 전달 중, 영상 정보 전달이 끊어진 상황이다. 이 경우 시스템은 재판부 PC로 해당 오류 상태를 알린다. |

#### Use Case Description

|  |  |
| --- | --- |
| **Use Case** | 법원TV 영상 전송 |
| **Actor** | A1. 재판부 PC, A2. 법원TV 방송시스템 |
| **Description** | 시스템은 재판부 PC를 통해 재판 방송 시작 요청을 받고, 재판 영상을 법원TV 방송 시스템에 전송한다. |
| **Stakeholders** | 재판부 PC, 법원TV 방송시스템, 법원, 법원 행정처 시스템 관리팀 |
| **Preconditions** | 해당 재판이 방송되어야 함으로 설정된 상태 |
| **Main Scenario** | (A1) 재판부 PC, (A2) 법원TV 방송시스템, (S) 시스템  1. (A1)는 (S)에게 방송 시작 요청을 한다.  2. (S)는 재판 영상 데이터를 (A2)로 송출 한다.  3. (A2)는 재판 영상 데이터 수신 시작 함을 (S)에게 알린다  3. (A1)는 (S)에게 방송 중단 요청을 한다.  4. (S)는 재판 영상 데이터 송출을 중단할 것임을 (A2)에게 알리고 송출을 중단한다.  5. (A2)는 재판 영상 송출 중단 확인을 (S)에게 알린다.  5. (S)는 (A2)로 영상 송출이 중단 되었음을 (A1)에게 보여준다. |
| **Alternative Scenario** | AF1. 영상 데이터 전송 중단  분기점: Main Scenario 3   1. (A2)는 (S)에게 영상 데이터에 오류가 있음을 확인하고 (S)에게 알린다. 2. (S)는 (A1)에 영상 데이터 전송에 오류가 생김을 알린다. |

#### 

#### System Sequence Diagram



|  |  |
| --- | --- |
| **System Interface / Operation** | **Description** |
| *startBroadcasting()* | 방송 시작 요청 |
| *broadcastingStatus()* | 방송 영상 송출 결과 수신   * OK * NOT\_OK * NORMAL\_END |
| *stopBroadcasting()* | 방송 중단 요청 |

### UC-04 영상 법정 연동

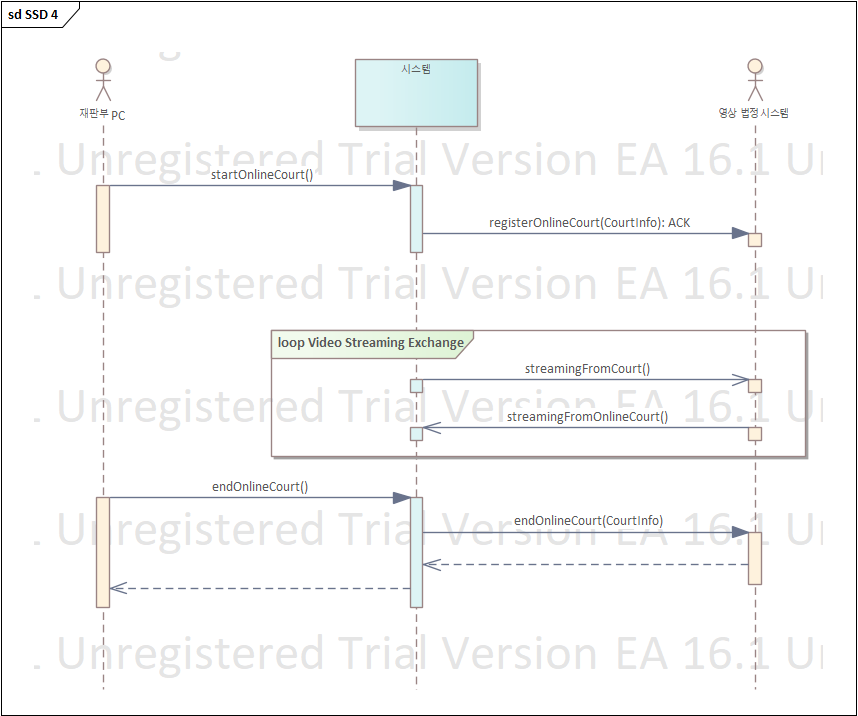
#### Scenario List

|  |  |
| --- | --- |
| **Scenario Title** | **Scenario Description** |
| 영상 법정 시작 | 재판부가 영상 법정을 시작하는 경우, 본 시스템과 영상법정 시스템은 각각 공유 영상을 상대 시스템에 전달 한다. |

#### Use Case Description

|  |  |
| --- | --- |
| **Use Case** | 영상 법정 시작 |
| **Actor** | A1. 재판부 PC, A2. 영상 법정 시스템 |
| **Description** | 재판이 영상 재판으로 진행될 때, 본 시스템과 영상법정은 각각 영상을 상대 시스템에 전달 한다. |
| **Stakeholders** | 재판부, 법원, 영상재판 운영지원단 |
| **Preconditions** | 해당 재판의 타입이 영상 재판으로 설정된 상태 |
| **Main Scenario** | (A1) 재판부 PC, (A2) 영상 법정 시스템, (S) 시스템  1. (A1)는 (S)에게 영상재판시작 요청을 한다.  2. (S)는 (A2)로 연결을 요청하고 영상 법정 시스템의 영상을 수신 할 수있게 등록한다.  3. (A2)는 (S)에게 연결이 성공 하였음을 알리고 시스템의 영상을 수신 할 수 있게 등록한다.  4. (A1)는 (S)에게 영상재판 중단 요청을 한다.  5. (S)는 영상재판을 중단할 것임을 (A2)에게 알리고 송출을 중단한다.  6. (A2)는 (S)에게 영상재판 중단 요청 확인을 알린다.  7. (S)는 (A1)에게 영상 재판이 종료 되었음을 알린다. |
| **Alternative Scenario** |  |

#### System Sequence Diagram



|  |  |
| --- | --- |
| **System Interface / Operation** | **Description** |
| *startOnlineCourt()* | 영상재판 시작 요청 기능 |
| *registerCourt()* | 재판 현장 영상을 받기 위한 등록 기능 |
| *streamingFromOnlineCourt()* | 영상법정으로 부터 영상재판의 영상을 받기위한 인터페이스 |
| *stopECourt()* | 영상재판 중단 요청 기능 |

### UC-05 자료 승인 요청

#### Scenario List

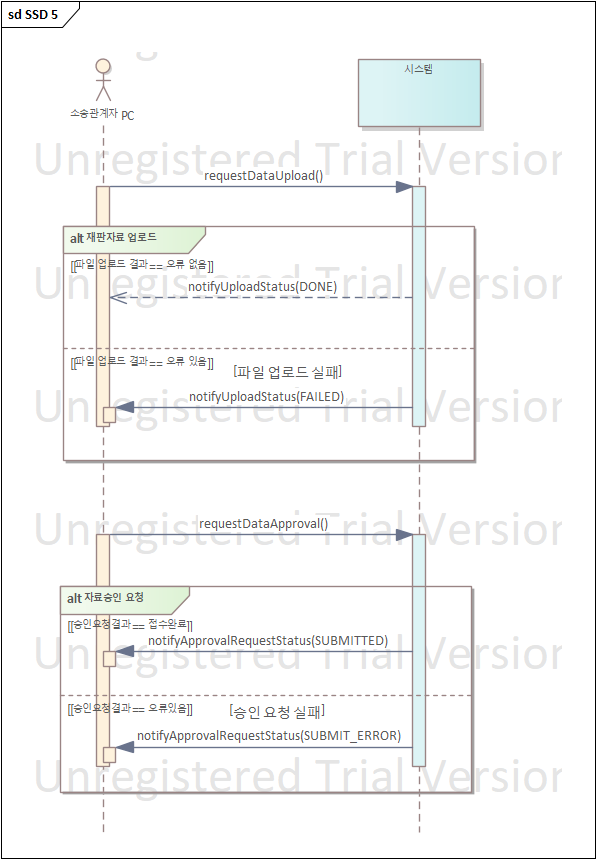
|  |  |
| --- | --- |
| **Scenario Title** | **Scenario Description** |
| 자료 승인 요청 | 소송 관계자는 소송 관계자 PC를 통해 자료를 업로드 하고, 해당 자료가 재판에서 사용될 수 있도록 자료 승인 요청을 한다. |
| 파일 업로드 실패 | 소송 관계자가 자료 승인 요청 하기 위해 자료를 시스템에 업로드를 할 때, 업로드가 실패한 경우이다. 이 경우 시스템은 소송 관계자 PC로 해당 실패 상황을 보여준다. |
| 승인 요청 실패 | 소송 관계자가 자료 승인 요청을 하였으나 실패한 경우이다. 이 경우 시스템은 소송 관계자 PC로 해당 실패 상황을 보여준다. |

#### Use Case Description

|  |  |
| --- | --- |
| **Use Case** | 자료 승인 요청 |
| **Actor** | A. 소송 관계자 PC |
| **Description** | 소송관계인 은 소송 관계자 PC를 통해서 자료를 업로드 한 후 승인 요청을 한다. |
| **Stakeholders** | (A) 소송 관계인 |
| **Preconditions** | 업로드 할 자료는 소송 관계자 PC에서 파일 접속이 가능한 상태  (PC에 자료 직접 저장 / 자료 저장된 USB를 소송 관계자 PC에 연결) |
| **Main Scenario** | (A) 소송 관계자 PC, (S) 시스템  1. (A) 소송 관계자 PC 에서 업로드할 파일을 업로드 한다.  2. (S) 는 (A)에게 업로드가 완료 됨을 알려준다.  3. (A) 는 업로드 된 자료 중 하나를 선택하여 (S)에게 승인 요청을 한다.  4. (S) 는 승인 요청이 정상적으로 접수 되었음을 (A)에게 알려주고 시나리오를 종료한다. |
| **Alternative Scenario** | AF1. 파일 업로드 실패  분기점: Main Scenario 2   1. (S)는 (A)에게 파일 업로드가 실패 했음을 알려주고 시나리오를 종료한다.   AF2. 승인 요청 실패  분기점: Main Scenario 4   1. (S)는 (A)에게 파일 업로드가 실패 했음을 알려주고 시나리오를 종료한다. |

#### 

#### System Sequence Diagram



|  |  |
| --- | --- |
| **System Interface / Operation** | **Description** |
| *requestDataUpload( )* | 자료 승인을 위한 파일을 소송관계자 PC에서 시스템으로 업로드 한다. |
| *requestDataApproval()* | 소송관계자는 시스템에 업로드한 파일의 승인을 요청 한다. |

### UC-06 재판부 자료 승인

#### Scenario List

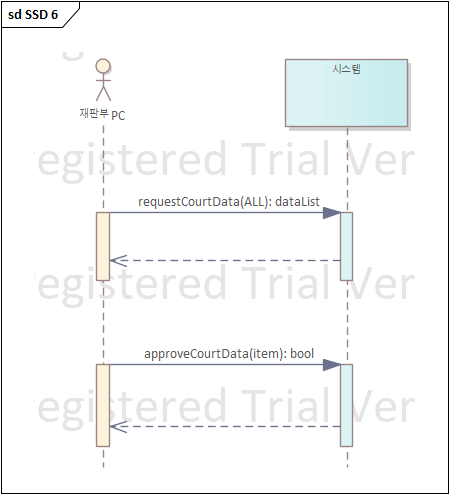
|  |  |
| --- | --- |
| **Scenario Title** | **Scenario Description** |
| 재판부 자료 승인 | 재판부는 재판부 PC를 통해 승인 요청 자료 리스트를 확인 하고 이를 승인 한다. |

#### Use Case Description

|  |  |
| --- | --- |
| **Use Case** | 재판부 자료 승인 |
| **Actor** | A. 재판부 PC |
| **Description** | 재판부는 재판부 PC를 통해 승인 요청된 자료 리스트를 확인 후 승인 할 수 있다. |
| **Stakeholders** | (A) 재판부 |
| **Preconditions** | 승인 요청된 자료가 있음 |
| **Main Scenario** | (A) 재판부 PC, (S) 시스템  1. (A) 는 (S)에 승인 요청된 자료 리스트 요청한다.  2. (S) 는 (A)에게 자료 리스트를 보여준다.  3. (A) 는 자료 중 하나를 선택하여 (S)에게 승인을 한다.  4. (S) 는 승인이 정상적으로 완료 되었음을 (A)에게 알려준다. |
| **Alternative Scenario** |  |

#### 

#### System Sequence Diagram



|  |  |
| --- | --- |
| **System Interface / Operation** | **Description** |
| *requestCourtData( )* | 재판 자료 리스트를 요청하는 기능 |
| *approveCourtData()* | 선택된 재판 자료를 승인하는 기능 |

### UC-07 법원 자료 업데이트

#### Scenario List

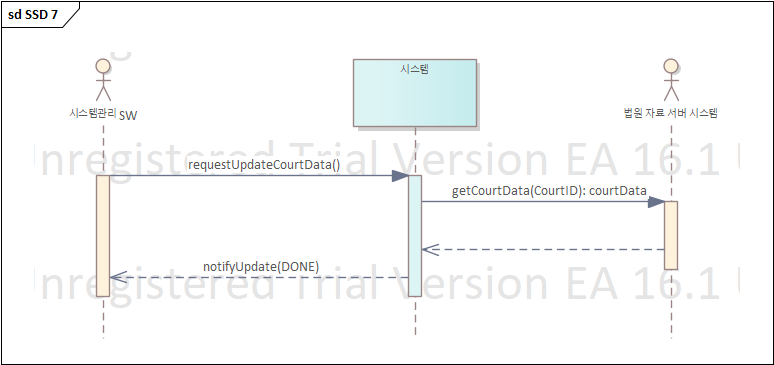
|  |  |
| --- | --- |
| **Scenario Title** | **Scenario Description** |
| 법원 자료 업데이트 | 시스템관리자는 시스템관리 SW를 통해 재판 시작 전에 법원 자료 서버 시스템에서 이전 재판 자료를 가져와 본 시스템을 업데이트 한다. |

#### Use Case Description

|  |  |
| --- | --- |
| **Use Case** | 법원 자료 업데이트 |
| **Actor** | A1. 법원 자료 서버 시스템, A2. 시스템관리 SW |
| **Description** | 시스템관리자는 시스템관리 SW를 통해 재판 시작 전에 법원 자료 서버 시스템에서 이전 재판 자료를 가져와 본 시스템을 업데이트 한다. |
| **Stakeholders** | 법원, 법원 행정처 시스템 관리팀 |
| **Preconditions** | 재판 진행이 확정 됨 |
| **Main Scenario** | (A1) 법원 자료 서버 시스템, (A2) 시스템관리 SW, (S) 시스템  1. (A2) 는 (S)에 재판 자료 업데이트를 요청한다.  2. (S) 는 (A1)에게 기존 재판 자료를 요청 한다.  3. (A1) 는 기존 재판 자료를 (S)에게 전달한다  4. (S)는 재판 자료가 업데이트 되었음을 (A2)에게 알리고 시나리오 종료한다. |
| **Alternative Scenario** |  |

#### 

#### System Sequence Diagram



|  |  |
| --- | --- |
| **System Interface / Operation** | **Description** |
| *requestUpdateCourtData( )* | 디지털 법정 재판의 기존 자료를 법원자료 서버 시스템의 데이터로 업데이트 요청 하는 기능 |

### UC-08 공용 영상 제어

#### Scenario List

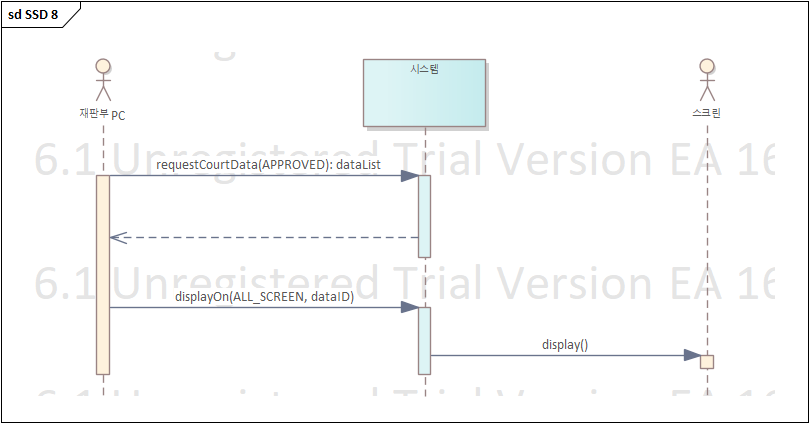
|  |  |
| --- | --- |
| **Scenario Title** | **Scenario Description** |
| 공용 영상 제어 | 재판의 공용 스크린에 영상 정보를 선택하여 보여주는 기능을 제어한다. |

#### Use Case Description

|  |  |
| --- | --- |
| **Use Case** | 공용 영상 제어 |
| **Actor** | A. 재판부 PC |
| **Description** | 시스템관리자는 시스템관리 SW를 통해 재판 시작 전에 법원 자료 서버 시스템에서 이전 재판 자료를 가져와 본 시스템을 업데이트 한다. |
| **Stakeholders** | 재판부 |
| **Preconditions** | 법정 내 모든 멀티미디어 입력 장치와 스크린 정상 연동 된 상태  승인된 재판 자료가 있는 상태 |
| **Main Scenario** | (A1) 재판부 PC, (S) 시스템, (A2) 스크린  1. (A1) 는 (S)에 승인된 자료 리스트 요청한다  2. (S) 는 (A1)에게 자료 리스트를 보여준다.  3. (A1) 는 공유할 스크린과 자료를 결정하여 (S)에 보여주도록 요청한다.  4. (S)는 (A2)에게 해당 자료를 보이도록 한다. |
| **Alternative Scenario** |  |

#### 

#### System Sequence Diagram



|  |  |
| --- | --- |
| **System Interface / Operation** | **Description** |
| *requestApprovedCourtData( )* | 승인된 자료 리스트 요청 |
| *displayOn()* | 어느 공용 스크린에 어떤 자료를 보여주도록 요청한다.   * ALL\_SCREEN, SCREEN\_NUMBER (스크린 디바이스) * data ID (Data 의 ID) |

### UC-09 시스템 연동 모니터링

#### Scenario List

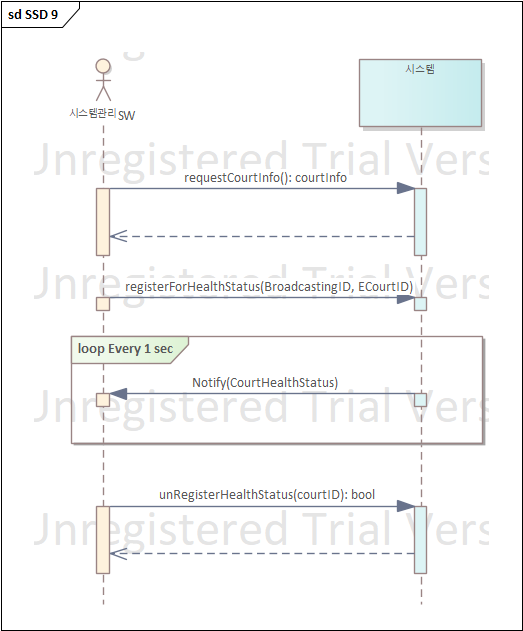
|  |  |
| --- | --- |
| **Scenario Title** | **Scenario Description** |
| 시스템 연동 모니터링 | 시스템관리SW는 본 시스템을 통해 외부 시스템(법원TV, 영상법정)과의 연동 기능에 대해서 시스템의 구간별 동작이 원활한지 확인할 수 있는 정보를 제공 받는다 |

#### Use Case Description

|  |  |
| --- | --- |
| **Use Case** | 시스템 연동 모니터링 |
| **Actor** | A 시스템관리 SW |
| **Description** | 재판의 기준 정보를 확인하고 해당 재판이 법원TV로 생중계 되거나, 영상법정으로 재판이 진행 되는 경우, 시스템의 구간별 기능 동작 여부를 모니터링 하는 기능 |
| **Stakeholders** | 법원 행정처 시스템 관리팀 |
| **Preconditions** | 재판이 방송을 위해 법원TV 시스템에 영상을 송출하고 있는 상태이거나 영상 재판으로 진행 중인 상태 |
| **Main Scenario** | (A) 시스템관리 SW, (S) 시스템  1. (A)는 (S) 로 현재 재판의 기준 정보를 요청한다.  2. (A) 는 (S)에 외부 연동 시스템의 강건성 데이터를 요청 한다.  3. (S) 는 (A)에게 시스템의 상태 정보를 정기적으로 받기 위해 subscribe 한다.  4. (S) 는 (A)에게 1초 마다 상태 정보를 notify 한다.  3. (A) 는 (S)에 시스템 강건성 데이터를 전송 중단을 요청 한다.  4. (S) 는 (A)에게 전송이 중단 되었음을 알려준다. |
| **Alternative Scenario** |  |

#### 

#### System Sequence Diagram



|  |  |
| --- | --- |
| **System Interface / Operation** | **Description** |
| *requestCourtInfo()* | 디지털 법정의 기준 정보 요청 기능 |
| *registerForHealthStatus()* | 디지털 법정의 실시간 운영 상태 정보를 받기 위해 등록 기능 |
| *unRegisterHealthStatus()* | 실시간 운영 상태 정보를 더이상 받지 않기 위한 등록 해제 기능 |

### UC-10 재판 정보 설정

#### Scenario List

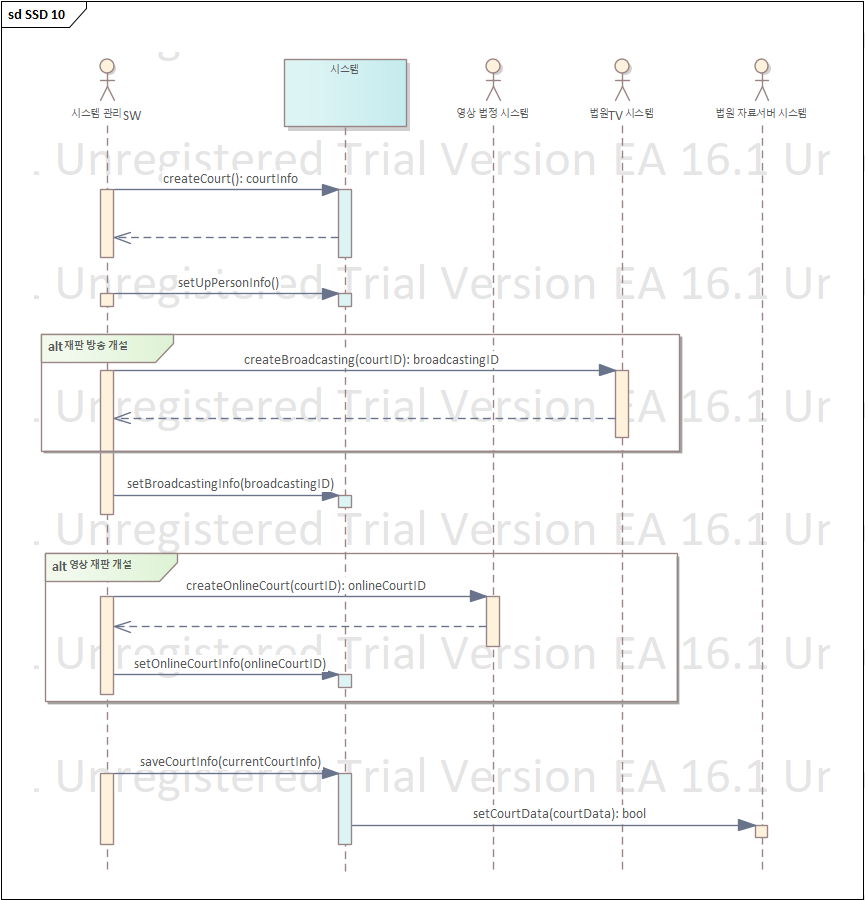
|  |  |
| --- | --- |
| **Scenario Title** | **Scenario Description** |
| 재판 정보 설정 | 디지털 법정의 기준 정보를 설정하는 기능 |
| 재판 방송 개설 | 디지털 법정을 방송하기위해 법원TV 시스템에 방송 개설 요청을 하는 기능 |
| 영상 재판 개설 | 디지털 법정을 영상 재판으로 개설 하기 위해 영상 법정 시스템에 개설 요청을 하는 기능 |

#### Use Case Description

|  |  |
| --- | --- |
| **Use Case** | 재판 정보 설정 |
| **Actor** | A1. 시스템관리 SW, A2. 영상 법정 시스템, A3. 법원TV 시스템 |
| **Description** | 시스템관리 SW는 법원 자료 서버를 통해서 재판에 필요한 기준 정보 및 기존 재판 정보를 확인 하여 디지털 법정의 재판 정보를 본 시스템에 설정하여 재판이 시작 될 수 있게 한다 |
| **Stakeholders** | 법원 행정처 시스템 관리팀, 영상 재판 운영 지원단, 법원TV |
| **Preconditions** | 재판부와 소송관계인이 확정된 상태  현재 재판의 이전 정보가 법원 자료 서버에 존재하는 상태 |
| **Main Scenario** | (A1) 시스템관리 SW, (S) 시스템, (A2) 영상 법정 시스템, (A3) 법원TV 시스템  1. (A1) 는 (S)에 재판 생성을 요청한다.  2. (A1) 는 (S)에 법원 자료 서버를 통해 받은 재판부와 소송관계인 정보를 입력한다.  3. (A1) 는 (S)에 재판 방송 정보를 설정 한다. (ECourtID/NA)  4. (A1) 는 (S)에 영상 재판 정보를 설정 한다. (ECourtID/NA)  5. (A1) 는 (S)에 기준 정보 저장을 요청 한다. |
| **Alternative Scenario** | AF2. 재판 방송 개설  분기점: Main Scenario 3   1. (A1) 는 (A3) 에게 재판 방송 개설을 요청하고 방송 ID를 부여받는다   AF2. 영상 재판 개설  분기점: Main Scenario 4   1. (A1) 는 (A2) 에게 영상 재판 개설을 요청하고 영상 재판 ID를 부여받는다. |

#### 

#### System Sequence Diagram



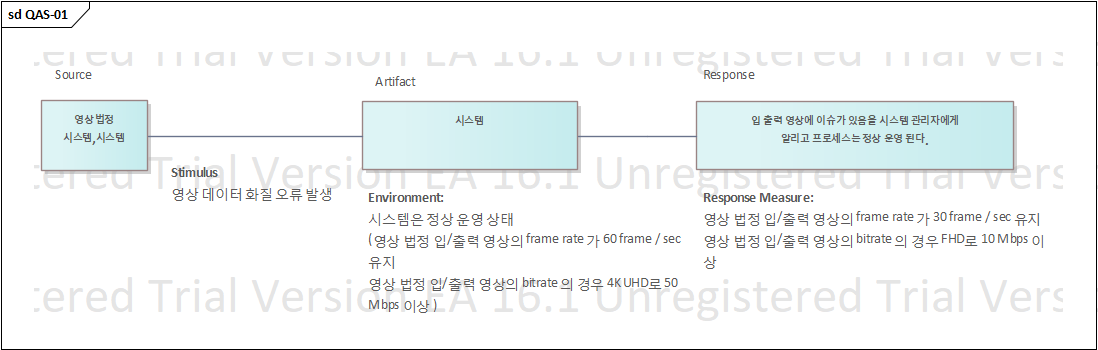
|  |  |
| --- | --- |
| **System Interface / Operation** | **Description** |
| *createCourt( )* | 디지털 법정을 생성하고 해당 ID를 리턴 하는 기능 |
| *setUpPersonInfo()* | 재판과 관련된 인력 정보 입력 (재판부, 소송관계인) 하는 기능 |
| *setBroadcastingInfo()* | 재판의 방송 정보 설정 기능 |
| *setOnlineCourtInfo()* | 재판의 영상재판 정보 설정 기능 |
| *saveCourtInfo()* | 현재 재판의 기준 정보 저장 기능 |

## Quality Attribute Scenario

### The QAS List

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Title** | **QA**  **Type** | **Priority** | | **Related**  **Use Case** | **System**  **Feature**  **ID** |
| **I** | **D** |
| QAS-01 | 영상재판 Latency 최소화 | Performance | 상 | 상 | UC-04 | SF-09  SF-10 |
| QAS-02 | 방청객 모바일 앱 사용자의 응답 속도 보장 | Performance | 상 | 중 | UC-01  UC-02  UC-03 | SF-02  SF-03  SF-06 |
| QAS-03 | ~~Device 변경 용이~~  각종 기능 추가 및 확장시 문제 없음 보장 | Extensibility | 중 | 하 | UC-03  UC-04  UC-08  UC-09  UC-10 | SF-06  SF-08  SF-09  SF-07  SF-13  SF-04  SF-05  SF-14  SF-15 |
| QAS-04 | 오류 없는 원활한 재판 진행 | Availability | 상 | 상 | UC-01  UC-02  UC-03  UC-04  UC-05  UC-06  UC-07  UC-08 | SF-02  SF-03  SF-06  SF-09  SF-10  SF-12  SF-11  SF-13 |
| QAS-05 | 시스템 장애 탐지 및 복구 | Availability | 중 | 중 | UC-09  UC-10 | SF-14  SF-15 |
| QAS-06 | 편리한 재판 자료 제어 | Usability | 상 | 중 | UC-05  UC-06 | SF-10  SF-12 |
| QAS-07 | 재판 자료 접근 제어 | Security | 중 | 중 | UC-01  UC-02  UC-05  UC-06  UC-08 | SF-01  SF-02  SF-03  SF-07  SF-10  SF-11  SF-13 |

### QAS-01 영상재판 Latency 최소화



시스템이 정상 운영 중일 때, 영상법정 및 해당 시스템을 통한 영상 딜레이 발생 시, 시스템 관리자에게 통지하고 프로세스는 Latency 우선 모드로 전환됩니다. 이때, 영상법정의 입/출력 영상은 Performance 측면에서 frame rate를 30 프레임, 비트레이트를 FHD 수준의 10Mbps 이상으로 유지해야 합니다.

|  |  |
| --- | --- |
| **QA Type** | Performance |
| **Description** | 영상 재판 진행 중 현장 영상을 실시간으로 전달 처리 하는 속도가 보장 되어야 한다. 그렇지 못할 경우 Latency 우선 모드로 전환 한다. |
| **Source of Stimulus** | 영상법정 시스템, 시스템 |
| **Stimulus** | 영상 딜레이 발생 |
| **Artifact** | 시스템 |
| **Environment** | 시스템은 정상 운영 상태  (영상 법정 입/출력 영상의 frame rate 가 60 frame / sec 유지  영상 법정 입/출력 영상의 bitrate 의 경우 4K UHD로 50 Mbps 이상 ) |
| **Response** | 입 출력 영상에 이슈가 있음을 시스템 관리자에게 Latency 우선 모드로 전환 후 네트워크 상태가 좋아지면 다시 복귀 시킨다 |
| **Response Measure** | 영상 법정 입/출력 영상의 frame rate 가 30 frame / sec 유지  영상 법정 입/출력 영상의 bitrate 의 경우 FHD로 10 Mbps 이상 |
| **Summary of QAS** | 시스템이 정상 운영 중일 때, 영상법정 및 해당 시스템을 통한 영상 딜레이 발생 시, 시스템 관리자에게 통지하고 프로세스는 Latency 우선 모드로 전환됩니다. 이때, 영상법정의 입/출력 영상은 Performance 측면에서 frame rate를 30 프레임, 비트레이트를 FHD 수준의 10Mbps 이상으로 유지해야 합니다. |

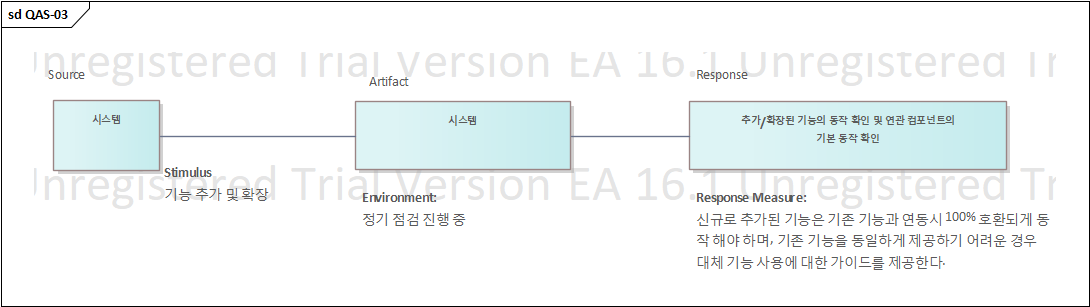
### QAS-02 방청객 모바일 앱 사용자의 응답 속도 보장



시스템이 정상 운영 중일 때, 스마트폰을 통한 재판 정보 요청에 대한 응답 시간은 3초 이내여야 합니다. 사용자는 요청 후 이 시간 안에 재판 정보 리스트를 볼 수 있어야 합니다.

|  |  |
| --- | --- |
| **QA Type** | Performance |
| **Description** | 재판 진행 시 방청객이 스마트폰을 통해 본 시스템의 모바일 앱을 사용할 때, 시스템의 응답속도를 보장해야 한다. |
| **Source of Stimulus** | 스마트폰 |
| **Stimulus** | 재판 정보 요청 |
| **Artifact** | 시스템 |
| **Environment** | 시스템은 정상 운영 상태 (CPU/메모리 50% 이상 확보, 네트워크 속도 10Gbps 이상) |
| **Response** | 스마트폰 사용자가 실시간으로 재판 정보 리스트를 볼 수 있어야 한다. |
| **Response Measure** | 스마트폰 사용자가 재판 정보 요청 후 3초 내에 정보 리스트가 보여야 한다. |
| **Summary of QAS** | 시스템이 정상 운영 중일 때, 스마트폰을 통한 재판 정보 요청에 대한 응답 시간은 3초 이내여야 합니다. 사용자는 요청 후 이 시간 안에 재판 정보 리스트를 볼 수 있어야 합니다. |

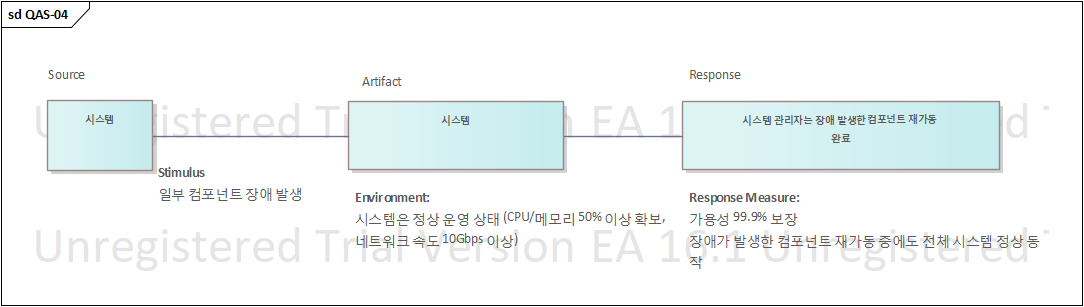
### QAS-03 각종 기능 추가 및 확장시 문제 없음 보장



정기 점검 진행 중 시스템에 외부 시스템과 연동되는 신규 기능 추가 및 확장시, 신규로 추가된 기능은 기존 기능과 100% 호환되어 동작 되어야 하며 기존 기능을 동일하게 제공하기 어려운 경우 대체 기능 사용에 대한 가이드를 제공한다.

|  |  |
| --- | --- |
| **QA Type** | Extensibility |
| **Description** | 본 시스템에 외부 시스템과 연동되는 특정 기능을 추가 하거나 확장 하더라도 기존 기능에 문제 없음을 보장한다. |
| **Source of Stimulus** | 시스템 |
| **Stimulus** | 기능 추가 및 확장 |
| **Artifact** | 시스템 |
| **Environment** | 정기 점검 진행 중 |
| **Response** | 추가/확장된 기능의 동작 확인 및 연관 컴포넌트의 기본 동작 확인 |
| **Response Measure** | ~~모든 장치 별 TC가 존재 해야 하며, Test Case는 실제 장치에서 수행할 수 있는 동작의 99% 가 테스트 가능해야 한다.~~  신규로 추가된 기능은 기존 기능과 연동시 100% 호환되게 동작 해야 하며, 기존 기능을 동일하게 제공하기 어려운 경우 대체 기능 사용에 대한 가이드를 제공한다. |
| **Summary of QAS** | 정기 점검 진행 중 시스템에 외부 시스템과 연동되는 신규 기능 추가 및 확장시, 신규로 추가된 기능은 기존 기능과 100% 호환되어 동작 되어야 하며 기존 기능을 동일하게 제공하기 어려운 경우 대체 기능 사용에 대한 가이드를 제공한다. |

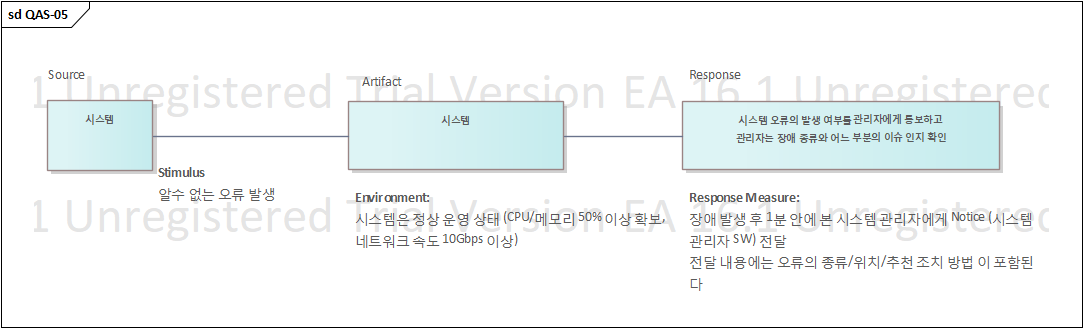
### QAS-04 오류 없는 원활한 재판 진행



시스템은 일부 컴포넌트의 서비스 불가 상태가 발생해도, 99.9%의 가용성을 유지해야 합니다. 또한, 장애 발생한 컴포넌트의 재가동 중에도 다른 컴포넌트는 정상적으로 동작해야 합니다.

|  |  |
| --- | --- |
| **QA Type** | Availability |
| **Description** | 시스템 일부에 장애가 발생하더라도 시스템은 정상 가동 해야 하며, 무중단 서비스 제공이 가능해야 한다. |
| **Source of Stimulus** | 시스템 |
| **Stimulus** | 시스템 일부 컴포넌트의 서비스 불가 상태 발생 |
| **Artifact** | 시스템 전체 |
| **Environment** | 시스템은 정상 운영 상태 (CPU/메모리 50% 이상 확보, 네트워크 속도 10Gbps 이상) |
| **Response** | 시스템 관리자는 문제 발생한 컴포넌트 재가동 완료 |
| **Response Measure** | 가용성 99.9% (재판 시작 부터 완료 시점까지)  장애 발생한 컴포넌트 재가동 중에도 다른 컴포넌트는 정상 동작 해야 한다. (재판 진행 중에 시스템 점검은 하지 않는다) |
| **Summary of QAS** | 시스템은 일부 컴포넌트의 서비스 불가 상태가 발생해도, 99.9%의 가용성을 유지해야 합니다. 또한, 장애 발생한 컴포넌트의 재가동 중에도 다른 컴포넌트는 정상적으로 동작해야 합니다. |

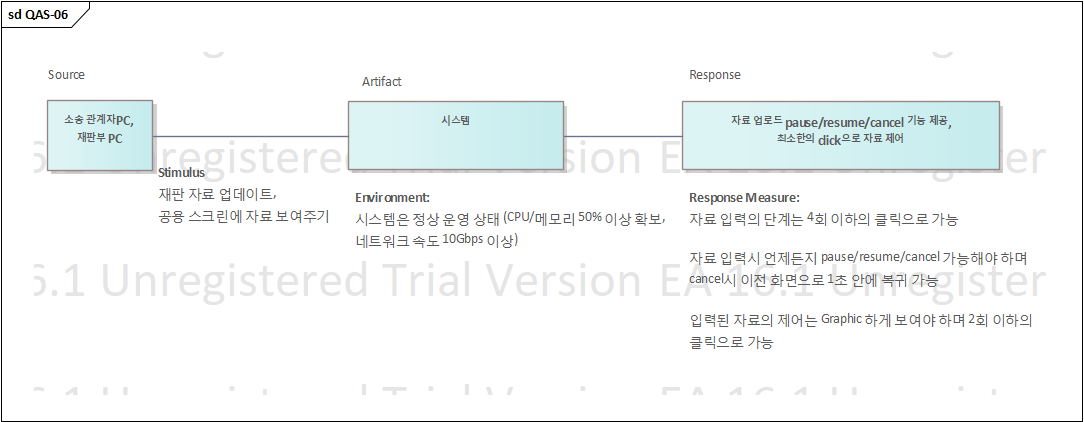
### QAS-05 시스템 장애 탐지 및 복구



시스템이 정상 상태에서 알 수 없는 오류가 발생하면, 해당 오류는 1분 이내에 시스템 관리자 소프트웨어에 자동으로 전달됩니다. 전달내용에는 오류의 종류, 위치 및 추천 조치 방법이 포함됩니다.

|  |  |
| --- | --- |
| **QA Type** | Availability |
| **Description** | 시스템 관리자는 디지털 재판 진행 중 장애 발생시 이를 감지할 수 있어야 하며 이를 통해 시스템 내의 어느 컴포넌트가 장애가 있는지 확인 할 수 있다. |
| **Source of Stimulus** | 시스템 |
| **Stimulus** | 알수 없는 오류 발생 |
| **Artifact** | 시스템 |
| **Environment** | 시스템은 정상 운영 상태 (CPU/메모리 50% 이상 확보, 네트워크 속도 10Gbps 이상) |
| **Response** | 시스템 오류의 발생 여부를 관리자에게 통보하고 관리자는 장애 종류와 어느 부분의 이슈 인지 확인 |
| **Response Measure** | 장애 발생 후 1분 안에 본 시스템 관리자에게 Notice (시스템 관리자 SW) 전달합니다. 전달 내용에는 오류의 종류/위치/추천 조치 방법 이 포함되어야 합니다. |
| **Summary of QAS** | 시스템이 정상 상태에서 알 수 없는 오류가 발생하면, 해당 오류는 1분 이내에 시스템 관리자 소프트웨어에 자동으로 전달됩니다. 전달내용에는 오류의 종류, 위치 및 추천 조치 방법이 포함됩니다. |

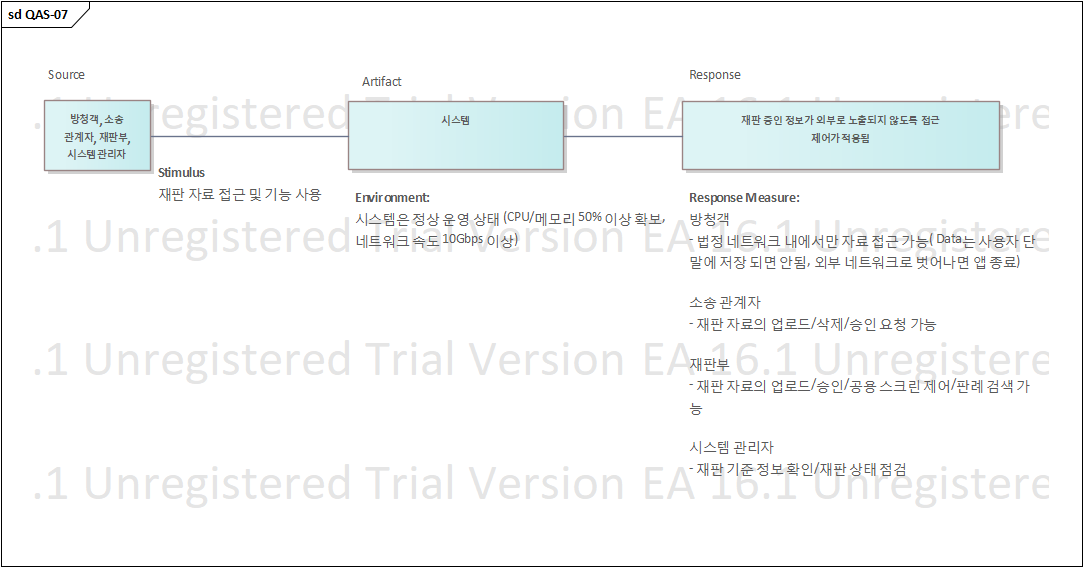
### QAS-06 편리한 재판 자료 제어



시스템이 정상 상태에서, 소송 관계자나 재판부가 재판 자료를 업데이트하거나 제어할 때, 자료 업로드를 중지하거나 다시 시작하거나 취소할 수 있어야 합니다. 또한, 사용자가 자료를 제어하는 과정을 최소한의 클릭으로 간편하게 수행할 수 있어야 합니다.

|  |  |
| --- | --- |
| **QA Type** | Usability |
| **Description** | 자료 입력시, 정확한 자료 입력이 가능해야 하며, 잘못 입력한 정보의 취소 및 복원이 가능해 한다. 또한 재판 자료는 seamless 하게 방청객이 볼 수있어야 하며, 공용 스크린에 보여주는 단계도 직관적이고 간단해야 한다. |
| **Source of Stimulus** | 소송 관계자 PC, 재판부 PC |
| **Stimulus** | 재판 자료 업데이트, 공용 스크린에 자료 보여주기 |
| **Artifact** | 시스템 |
| **Environment** | 시스템은 정상 운영 상태 (CPU/메모리 50% 이상 확보, 네트워크 속도 10Gbps 이상) |
| **Response** | 자료 업로드 pause/resume/cancel 기능 제공, 최소한의 click으로 자료 제어 |
| **Response Measure** | 자료 입력은 4회 이하의 클릭으로 가능해야 합니다.  또한, 자료 입력 중 언제든지 일시 정지, 다시 시작 또는 취소할 수 있어야 하며, 취소 시 이전 화면으로 1초 안에 복귀해야 합니다.  방청객은 seamless 하게 재판 자료를 받을 수 있어야 합니다.  입력된 자료의 제어는 그래픽으로 표시되어야 하며, 2회 이하의 클릭으로 가능해야 합니다.  이를 통해 본 시스템의 사용자는 효율적으로 작업을 할 수 있어야 합니다 |
| **Summary of QAS** | 시스템이 정상 상태에서, 소송 관계자나 재판부가 재판 자료를 업데이트하거나 제어할 때, 자료 업로드를 중지하거나 다시 시작하거나 취소할 수 있어야 합니다. 또한, 사용자가 자료를 제어하는 과정을 최소한의 클릭으로 간편하게 수행할 수 있어야 합니다. |

### QAS-07 재판 자료 접근 제어



시스템이 정상 운영 중인 상태에서, 시스템은 법정 네트워크 내에서만 자료 접근을 허용하며, 방청객은 화면 캡쳐가 차단되고 데이터는 장치에 저장되지 않습니다. 소송 관계자 및 재판부는 자료 업로드/삭제/승인/제어가 가능하며, 시스템 관리자는 재판 상태를 확인할 수 있습니다. 이는 환경이 정상인 시스템에서 법원의 보안 점검 기준을 충족하여 재판 자료의 보안을 보장합니다.

|  |  |
| --- | --- |
| **QA Type** | Security |
| **Description** | 본 시스템을 사용하는 모든 사용자는 자신이 가진 접근 권한 이상의 자료를 열람 하거나 변경 할 수 없고, 기능의 사용도 접근 권한에 맞게 제한된다. |
| **Source of Stimulus** | 방청객, 소송 관계자, 재판부, 시스템 관리자 |
| **Stimulus** | 재판 자료 접근 및 기능 사용 |
| **Artifact** | 시스템 |
| **Environment** | 시스템은 정상 운영 상태 (CPU/메모리 50% 이상 확보, 네트워크 속도 10Gbps 이상) |
| **Response** | 재판 중인 정보가 외부로 노출되지 않도록 접근 제어가 적용됨 |
| **Response Measure** | 방청객:   * 법정 네트워크 내에서만 자료 접근 가능하며, 데이터는 사용자 단말에 저장되지 않고 외부 네트워크로 벗어나면 앱이 종료됩니다. 또한, 재판 중에는 화면 캡쳐가 차단됩니다.   소송 관계자:   * 재판 자료의 업로드, 삭제, 승인 요청이 가능합니다.   재판부:   * 재판 자료의 업로드, 승인, 공용 스크린 제어, 판례 검색이 가능합니다.   시스템 관리자:   * 재판 기준 정보를 확인하고 재판 상태를 점검할 수 있습니다.   이를 통해 재판 자료의 보안 기준을 충족하여 법원의 정기 보안 점검을 통과할 수 있습니다. |
| **Summary of QAS** | 시스템이 정상 운영 중인 상태에서, 시스템은 법정 네트워크 내에서만 자료 접근을 허용하며, 방청객은 화면 캡쳐가 차단되고 데이터는 장치에 저장되지 않습니다.  소송 관계자 및 재판부는 자료 업로드/삭제/승인/제어가 가능하며, 시스템 관리자는 재판 상태를 확인할 수 있습니다. 이는 환경이 정상인 시스템에서 법원의 보안 점검 기준을 충족하여 재판 자료의 보안을 보장합니다. |

## Constraint

### Business Constraint List

|  |  |  |
| --- | --- | --- |
| **ID** | **Title** | **Description** |
| NA | NA | NA |

### Technical Constraint List

|  |  |  |
| --- | --- | --- |
| **ID** | **Title** | **Description** |
| NA | NA | NA |

# Architecture Design & Evaluation

## Candidate Designs per QA

### Candidate Design List

|  |  |  |  |
| --- | --- | --- | --- |
| **QA** | **QAS** | **Candidate Design** | **Candidate Design Approach (CDA)** |
| QA1:  Performance | QAS-01  QAS-02 | QA1\_CD | QA1\_CD\_CDA-01 : HLS 방식 + Repository  QA1\_CD\_CDA-02 : Web RTC 방식 + in memory caching  QA1\_CD\_CDA-03 : Web RTC 방식 + DB Master-Slave + in memory caching |
| QA2:  Extensibility | QAS-03 | QA2\_CD | QA2\_CD\_CDA-01 : Microservice  QA2\_CD\_CDA-02 : Broker  QA2\_CD\_CDA-03 : Dispatcher |
| QA3:  Availability | QAS-04  QAS-05 | QA3\_CD | QA3\_CD\_CDA-01 : Server-Client + Real-time monitoring  QA3\_CD\_CDA-02 : Multi-Tier + Master-Slave + Ping/Echo + Heartbeat  QA3\_CD\_CDA-03 : 3-Tier + Master-Slave + Ping/Echo + Heartbeat |
| QA4:  Usability | QAS-06 | QA4\_CD | QA4\_CD\_CDA-01 :Mobile application + Web Application  QA4\_CD\_CDA-02 : Mobile application(PAC) + Rich Internet Application  QA4\_CD\_CDA-03 : Mobile application(MVC) + Web Application + Data Preview |
| QA5:  Security | QAS-07 | QA5\_CD | QA5\_CD\_CDA-01 : VPN + Identify actors + Change default settings + Encrypt data(콘텐츠 레벨)  QA5\_CD\_CDA-02 : VPN + Authenticate actors + Change default settings + Encrypt data(채널 레벨)  QA5\_CD\_CDA-03 : VPN + Encrypt data(콘텐츠 + 채널 레벨) |

### QA1: Performance

#### Design Goal

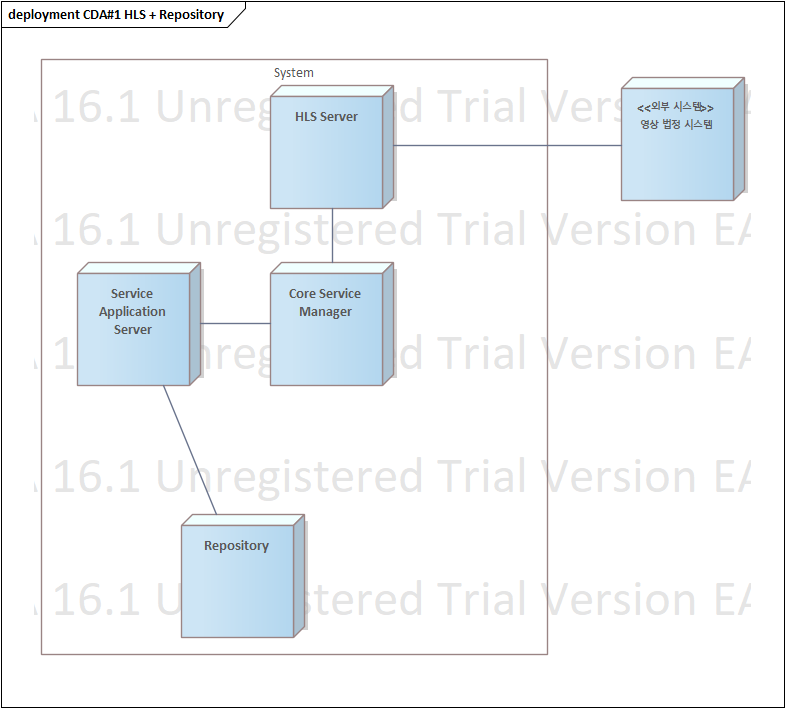
시스템은 영상 재판 진행 시 영상 Latency가 최소화 되기를 원하며, 방청객이 모바일 앱을 사용할 때도 빠른 반응 속도로 사용할 수 있길 원한다. 영상 데이터에 대한 성능 확보 방법에 대해 확인시 **HLS 방식과 Web RTC 방식**, 그리고 모바일 앱에서 서버의 재판 자료를 가져올 때 DB의 저장 방식을 일반 **Repository 방식**과 **in memory를 사용한 Caching** 방법 그리고 **DB의 Master-Slave** 구조 사용 방식을 비교하고자 한다.

다음은 Performance 측면에서의 3가지 Candidate Design Approach 이다.

#### Candidate Design Approach List

#### CDA #1 Description: HLS 방식 + Repository

|  |  |
| --- | --- |
| QA1\_CD\_CDA-01 | HLS 방식 + Repository |
| **Description** | **ㅁ HLS(HTTP Live Streaming) 설명**  인덱싱  - 재생 순서와 각 청크의 URI를 담은 m3u8 확장자의 플레이리스트 파일을 생성합니다.  다중 비트레이트 지원  - 원본 비디오를 여러 비트레이트와 해상도로 인코딩합니다.  별도 플레이리스트를 통해 다양한 스트림 품질 제공이 가능합니다.  HTTP 기반 전송  - 청크 파일과 플레이리스트를 일반 HTTP 웹 서버로 전송합니다.  일반 웹서버나 CDN을 활용할 수 있습니다.  적응형 비트레이트 스트리밍  - 클라이언트가 현재 네트워크 상황에 맞는 적절한 품질 스트림을 선택합니다.  청크 단위로 비트레이트를 전환(bitrate switching)할 수 있습니다.  **ㅁ Repository 설명**  Architecture Catalog 중 Repository의 형태로 Data Centered Architecture 처럼 동작 한다. Database로써의 기본 기능을 제공할 수 있다.  Data Storage는 Passive 한 반면, Computational Process 들이 Active 하게 동작한다. |
| **Pros** | **ㅁ HLS 장점**  - 방화벽과 프록시 서버 등을 쉽게 통과할 수 있다.  - 비디오 시작이 빠르고 버퍼링이 적다.  - 모바일과 데스크톱 환경 모두에서 지원된다.  **ㅁ Repository 장점**  - 다른 Data Storage 구성 방법 보다 비교적 간단하다. |
| **Cons** | **ㅁ HLS 단점**  - 지연시간(Latency)  - **5~30초**  - 플레이어 개발 필요(HTML5 Video, 플러그인 등)  - **MPEC 계열 코덱만 지원**  **ㅁ Repository 단점**  - Storage의 File을 직접 Access 해야 하며 이 때 성능은 Repository의 성능을 따른다. (더 나은 부분을 기대하기 어려움) |

****

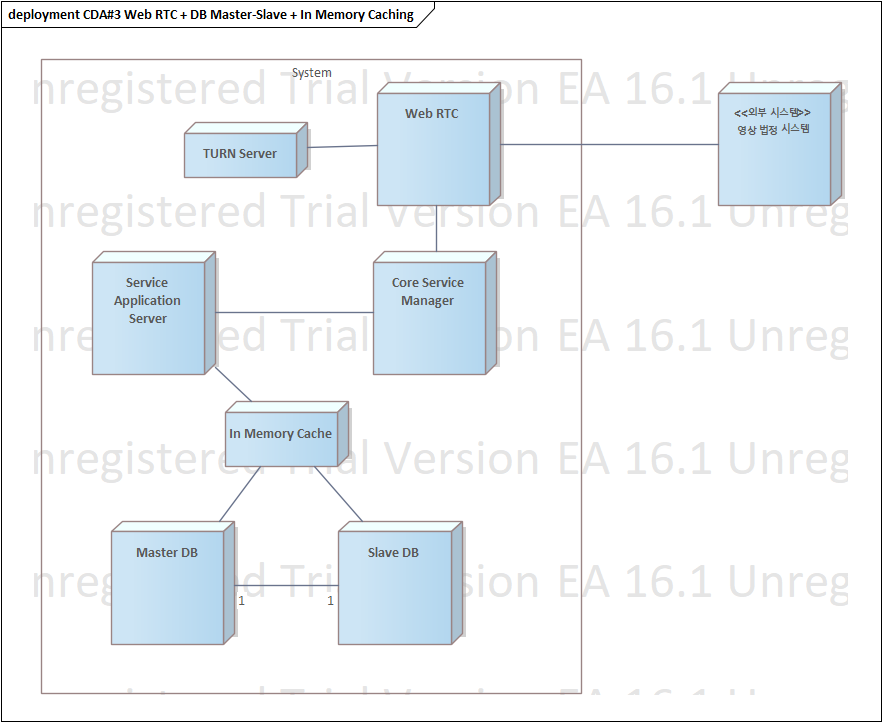
#### CDA #2 Description: Web RTC 방식 + In Memory Caching

|  |  |
| --- | --- |
| QA1\_CD\_CDA-02 | Web RTC 방식 적용 + in memory caching |
| **Description** | **ㅁ Web RTC 설명**  WebRTC(Web Real-Time Communication)는 웹 브라우저에서 실시간 미디어 통신을 가능케 하는 오픈소스 기술로 매우 낮은 지연시간의 실시간 스트리밍이 가능하고 간단한 구현이 가능하다.  **ㅁ In Memory Caching 설명**  **기본 Data Storage도 사용**하며 **이에 저장된 data중 일부를 Service Server의 DDR Memory에 상주**시켜서 database read/write에 필요한 I/O time을 최소화 시키는 방식이다. |
| **Pros** | **ㅁ Web RTC 방식 장점**  매우 낮은 지연시간(100~400ms)  - 실시간 화상 통신, 스트리밍에 최적화  - RTSP 등 기존 스트리밍 프로토콜보다 훨씬 낮은 지연  브라우저 기반 네이티브 지원  - 별도 플러그인이나 앱 설치 없이 웹에서 바로 스트리밍 가능  - 크로스 플랫폼 지원 (웹/모바일/데스크톱)  보안 및 방화벽 통과 지원  - SRTP로 미디어 암호화, DTLS로 키 교환 등 보안성 확보  - STUN, TURN 등으로 NAT, 방화벽 문제 해결  다양한 코덱 지원  - VP8, VP9, H.264 등 다양한 영상/음성 코덱 지원  **ㅁ In Memory Caching 장점**  - Caching이 되어 있는 경우 Data Storage의 read/write에 필요한 I/O time이 들지 않기 때문에 Data 사용을 위한 속도가 빨라짐 |
| **Cons** | **ㅁ Web RTC 방식 단점**  브라우저 제한  - 브라우저 벤더사의 WebRTC 지원 정책에 종속적  TURN 서버 비용  - 방화벽 환경에서는 **TURN 서버가 필수적이라 비용 발생**  동영상 스트리밍에 일부 제약  - 현재는 라이브/실시간 스트리밍에 최적화  **- VOD 스트리밍에는 아직 부족한 부분 있음**  브라우저 외 환경에서는 지원 미약  - 브라우저가 아닌 네이티브 앱에서의 WebRTC 활용 제한적  상대적으로 새로운 기술  - 안정성, 상용화 수준에서 검증 필요  **ㅁ In Memory Caching 단점**  - 기본 **Data Storage에 부가적으로** DDR Memory로 Caching 된 data를 추가로 가지고 있는 것이라 비용 발생  - Caching 되어 있지 않은 data를 사용하려 할 경우 Data Storage 접근이 필요하여 I/O time 발생함 |



#### CDA #3 Description: Web RTC 방식 + DB Master-Slave + In Memory Caching

|  |  |
| --- | --- |
| QA1\_CD\_CDA-03 | Web RTC 방식 + DB Master-Slave + in memory caching |
| **Description** | ㅁ Web RTC 설명   * CDA #2 와 동일   ㅁ In Memory Caching 설명   * CDA #2 와 동일   **ㅁ DB Master-Slave 설명**  DataBase를 Main 장치와 이를 replication 한 Slave로 구성한다. Main에서는 Write 권한을 가진 사용자가 data를 변경/추가/업데이트를 할 수 있으며, read도 가능하다. Slave는 일반적인 상태에서는 read만 가능하게 한다. |
| **Pros** | ㅁ Web RTC 방식 장점   * CDA #2 와 동일   ㅁ In Memory Caching 장점   * CDA #2 와 동일   **ㅁ DB Master-Slave 장점**  Write 권한을 가진 소송 관계인 및 재판부, 시스템 관리팀은 비교적 인원이 고정되어 있으며 적다. Read 권한을 가진 방청객은 최대 100명까지 가능하기 때문에 상대적으로 많은 리소스를 필요로 한다. 따라서 Master - Slave 방식으로 운영시 Read를 허용하는 리소스가 더욱 많기 때문에 빠르게 Read 기능이 가능하게 할 수 있다. |
| **Cons** | ㅁ Web RTC 방식 단점   * CDA #2 와 동일   ㅁ In Memory Caching 단점   * CDA #2 와 동일   **ㅁ DB Master-Slave 단점**   * 하나의 Main DB만 가지는 것 보다 물리적인 장비가 필요하여 비용 발생 * Master 와 Slave 간의 Data Sync 이슈 해결 필요 |

****

#### Decision and Rationale

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Performance* | | **Analysis** | QA1\_CD\_CDA-01 : HLS 방식 + Repository | QA1\_CD\_CDA-02 : Web RTC 방식 적용 + in memory caching | QA1\_CD\_CDA-03 : Web RTC 방식 + DB Master-Slave + in memory caching  **(Selected)** |
| **ID** | **Title** |
| QAS-01 | 영상재판 Latency 최소화 | Pros | (+) HLS 방식 사용시 HTTP 프로토콜 사용으로 기존 웹 인프라 사용 가능하며, 비디오 시작이 빠르고 버퍼링이 적음, VPN 환경이지만 방화벽과 프록시 서버 등을 쉽게 통과 가능 | (++) Web RTC 사용시 매우 낮은 지연시간으로 실시간 화상 통신에 최적화  별도 플러그인이나 앱 설치 없이 웹에서 바로 스트리밍 가능 | (++) Web RTC 내용은 CDA-02와 동일 |
| Cons | (-) 청크 크기 감소, 저지연 HLS, CDN 최적화 등의 기법을 추가로 사용하여 지연시간 극복이 어느정도 가능하나 품질저하나 비용증가 등이 있을 수 있음 | (-) Web RTC 사용시 방화벽 환경에서 TURN 서버가 필수적으로 비용발생  브라우저 중심이라, 향후 앱으로 확장시 추가 개발 필요 | (-) Web RTC 내용은 CDA-02와 동일 |
| QAS-02 | 방청객 모바일 앱 사용자의 응답 속도 보장 | Pros | (0) 비교적 간단하게 구현 가능 | (+) in memory caching 사용시 DB에 직접 접근 하는 빈도가 줄어들어 응답 속도 향상 | (++) in memory caching 방법과 DB의 Master-Slave 구조 도입을 같이 적용시 방청객의 숫자가 늘어나도 접근 가능한 DB가 하나 더 있기 때문에 상대적으로 더 빠른 응답 속도를 보장 |
| Cons | (0) 기본적인 Database storage 기능 외에 부가적인 성능 향상을 기대하기 어려움 | (-) 부가적이고 더 빠른 장치 사용으로 인한 비용 발생 | (-) 개발이 복잡해지고, 부가적인 장치 사용으로 인한 비용 발생 |

Candidate Design:

|  |  |  |  |
| --- | --- | --- | --- |
| **QA** | **QAS** | **CD** | **Description** |
| QA1:  Performance | QAS-01,  QAS-02 | QA1\_CD\_CDA-03 : Web RTC 방식 + DB Master-Slave + in memory caching | Web RTC를 도입하여 Latency 최적화된 영상 법정 품질을 제공하고, In memory caching 과 DB Master-Slave 방식을 통해 방청객이 모바일 앱을 통해 재판 정보를 가져올 때 최적의 응답 속도를 보장할 수 있도록 한다. |

### 

### QA2: Extensibility

#### Design Goal

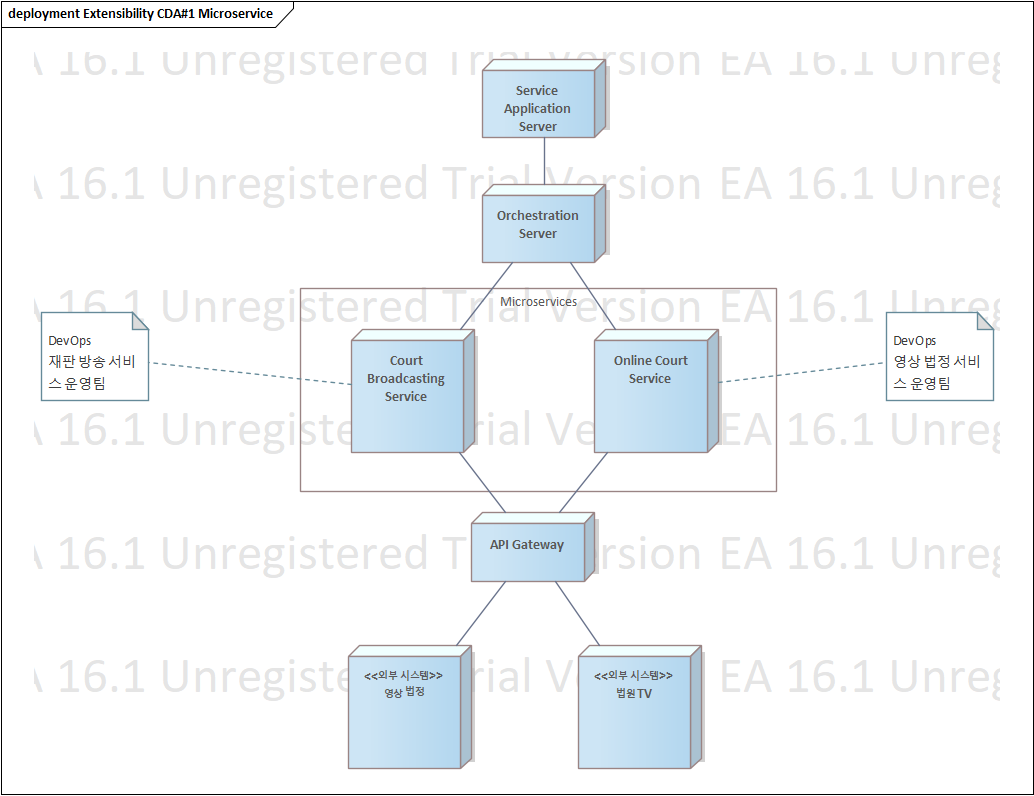
시스템은 외부 시스템과의 연동되는 기능의 확장 및 신규 추가 시 기존 기능과 100% 호환되어 동작 되어야 하며 기존 기능의 경우 동일하게 제공하기 어려우면 대체 기능에 대한 사용 가이드가 제공 되어야 한다. 이를 위해서 **Deployment Pattern**을 조사한 결과 Distributed architecture에 관련 내용이 있었고, 그 중에 **Microservice, Broker, Dispatcher** Architecture를 적용하여 Extensibility를 확보 하고자 한다.

다음은 Extensibility 측면에서의 3가지 Candidate Design Approach이다.

#### Candidate Design Approach List

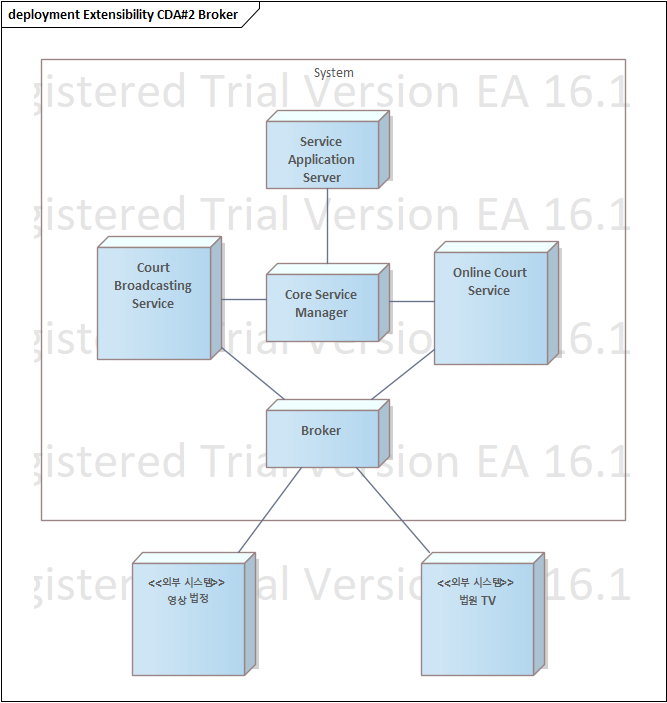
#### CDA #1 Description: Microservice

|  |  |
| --- | --- |
| QA2\_CD\_CDA-01 | Microservice |
| **Description** | Microservice 아키텍처는 기능별 분리 및 독립적 배포가 가능하며 새로운 기능 추가나 기존 기능 수정이 용이한 방식이다.  이를 디지털 법정에 도입하여 확장성을 최대한 확보하여 향후 서비스 운영에 유연하게 대응할 수 있도록 한다. |
| **Pros** | 기능별 분리 및 독립적 배포가 가능하여 새로운 기능 추가나 기존 기능 수정이 용이하다.  각 서비스마다 가장 적합한 기술 스택을 선택할 수 있어 기술 부채를 줄일 수 있다.  특정 서비스에 장애가 발생해도 다른 서비스에는 영향을 주지 않아 가용성이 높다. |
| **Cons** | 분산 시스템의 복잡성이 증가하여 모니터링, 로깅, 분산 추적 등 관리 부담이 크다.  서비스 간 통신 오버헤드가 발생하고, 데이터 일관성 유지가 어려울 수 있다. |



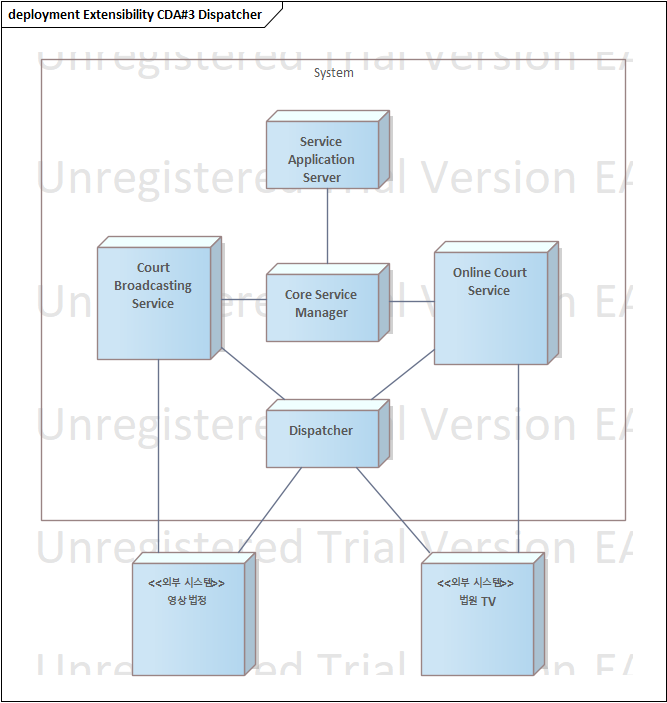
#### CDA #2 Description: Broker

|  |  |
| --- | --- |
| QA2\_CD\_CDA-02 | Broker |
| **Description** | 디지털 법정과 연동되는 외부 시스템 사이에 브로커를 두어 메시지를 라우팅하고 중재하는 방식이다.  이를 통해 시스템 간 직접적인 커플링을 낮추고 브로커를 통해야만 기능할 수 있도록 한다. |
| **Pros** | (+) 시스템이 독립적으로 동작할 수 있는 기반을 마련하고 일부 컴포넌트에 장애가 발생해도 메시지 손실 없이 처리 할 수 있다.  뿐만 아니라, 디지털 법정과 외부 시스템의 운영이 동시에 진행시 **중앙 통제가 수월**하면서도, **시스템간 결합도를 낮출** 수 있다.   * 각 시스템이 서로를 모르고 브로커를 통해서 간접적으로 통신하므로 결합도가 낮아진다. |
| **Cons** | (-) 브로커가 중앙 집중식 컴포넌트이기 때문에 single point of failure가 될 수 있으며, 브로커의 구현과 운영이 복잡하다.  또한 영상법정 시스템과 법원TV의 경우 생방송이나 실시간 미팅이 중요한데, 브로커를 통해야만 시스템간 연동이 가능해 **성능면에서 이슈**가 있을 수 있다. |



#### CDA #3 Description: Dispatcher

|  |  |
| --- | --- |
| QA2\_CD\_CDA-03 | Dispatcher |
| **Description** | Dispatcher 방식은 Broker의 장점을 가져오면서 그와 동시에 디지털 법정 시스템에 꼭 필요한 성능적인 요소도 놓치지 않기 위한 방법이다.  Broker와 동일하게 외부 시스템 연동시 Dispatcher 컴포넌트를 통해서 접근 시도를 하게 되지만, 실제 동작은 기능별 컴포넌트 (디지털 법정의 경우 Court Broadcasting 기능과 Online Court 기능) 와 직접 통신할 수 있게 하여 병목현상도 해결할 수 있다.  이를 통해 디지털 재판 운영, 영상 법정, 법원TV와 같은 서비스를 개별 모듈화 시켜 독립적으로 확장 가능하게 하며 향후 새로운 기능 모듈을 추가하기 용이하게 한다. |
| **Pros** | (++) 서비스 모듈화를 통해 서로 다른 기능을 개별 모듈로 분리하여 독립적으로 확장 가능 하며 추가도 용이함  기능 격리를 통해 특정 기능에 문제가 발생해도 다른 모듈에는 영향을 주지 않게 함  Broker 대비, Dispatcher 구조 사용시 실제 기능 동작은 기능별 컴포넌트와 직접 통신이 가능하기 때문에 **실시간 영상이 필수적인 영상 법정 기능 구현에 용이 함** |
| **Cons** | (-) 서비스 모듈이 증가하면 디스패처 로직이 복잡해 질 수 있음, Dispatcher의 확장성이 전체 시스템 확장성과 연관됨 |

****

#### Decision and Rationale

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Extensibility* | | **Analysis** | QA2\_CD\_CDA-01 : Microservice | QA2\_CD\_CDA-02 : Broker | QA2\_CD\_CDA-03 : Dispatcher  **(Selected)** |
| **ID** | **Title** |
| QAS-03 | 각종 기능 추가 및 확장시 문제 없음 보장 | Pros | (++) 서비스 확장성 | (+) 서비스 확장성, 서비스간 연동 통제 용이 | (++) 서비스 확장성, 서비스간 연동 통제 용이, 실시간 영상 법정 기능 구현시 성능 이슈 발생 가능성이 상대적으로 낮음(Broker 대비) |
| Cons | (--) 중앙 통제 어려움, 시스템간 연동시 시스템 오버헤드 발생 | (-) 개발 복잡, 실시간 영상 법정 기능 구현시 성능 이슈 발생 가능성  Broker의 Single point of failure 가능성 있음 | (-) Dispatcher의 Single point of failure 가능성 있음 |

Candidate Design:

|  |  |  |  |
| --- | --- | --- | --- |
| **QA** | **QAS** | **CD** | **Description** |
| QA2:  Extensibility | QAS-03 | *QA2\_CD\_CDA-03 : Dispatcher* | 순수 서비스 확장성만 검토시 Microservice가 용이하게 보일 수 있으나, 디지털 법정이라는 환경을 고려시 기존 디지털 재판 운영 서비스와 연동이 제대로 보장되는 것이 중요하며 이 때 오버헤드가 발생 하는 것은 최소화 해야 한다.  또한 영상법정 서비스와 같이 실시간성을 요하는 데이터 연동이 필요한 것을 고려시 시스템간 기능 연동시 connection 단계 이후 부터는 모듈 별 컴포넌트와 직접 통신이 가능해야 서비스 성능 확보에도 문제가 없을 것으로 보인다.  이와 같은 점들을 고려시 최종적으로는 Dispatcher 패턴을 채택하여 시스템을 구조화 하고 해당 패턴 채택시 발생 가능한 이슈는 향후 추가적인 architecture style 이나 tactic을 적용하여 극복하도록 하겠다. |

### QA3: Availability

#### Design Goal

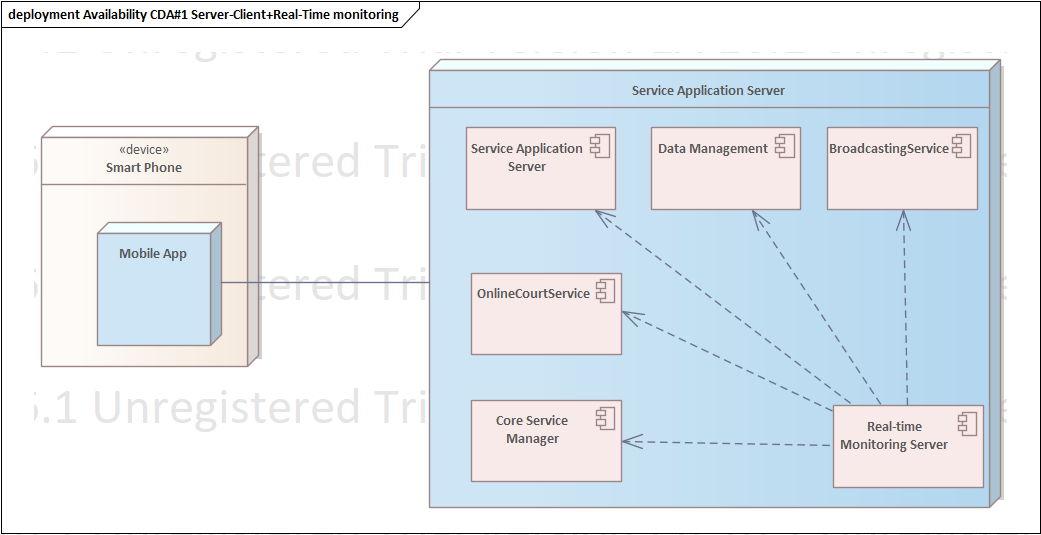
시스템은 사용자들에게 중단 없는 서비스를 제공해야 한다. 그리고 오류가 발생하면 이를 탐지할 수 있어야 하며, 시스템 관리자에게 알려주어 최대한 빨리 복구 시킬 수 있어야 한다. 따라서 가용성을 보장할 수 있는 **Deployment Pattern**을 조사한 결과 **Distributed architecture**에 관련 내용이 있었고, 그 중 **Server-Client, 4 Tier, 3 Tier architecture** 를 candidate로 두고 **장애 탐지 및 복구**를 위해서는 **Real-time monitoring, Master-Slave + Ping/Echo 적용 방식, 전체 시스템의 Heartbeat 모니터링**을 조합하여 Availability를 확보하고자 한다.

다음은 Availability 측면에서의 3가지 Candidate Design Approach 이다.

#### Candidate Design Approach List

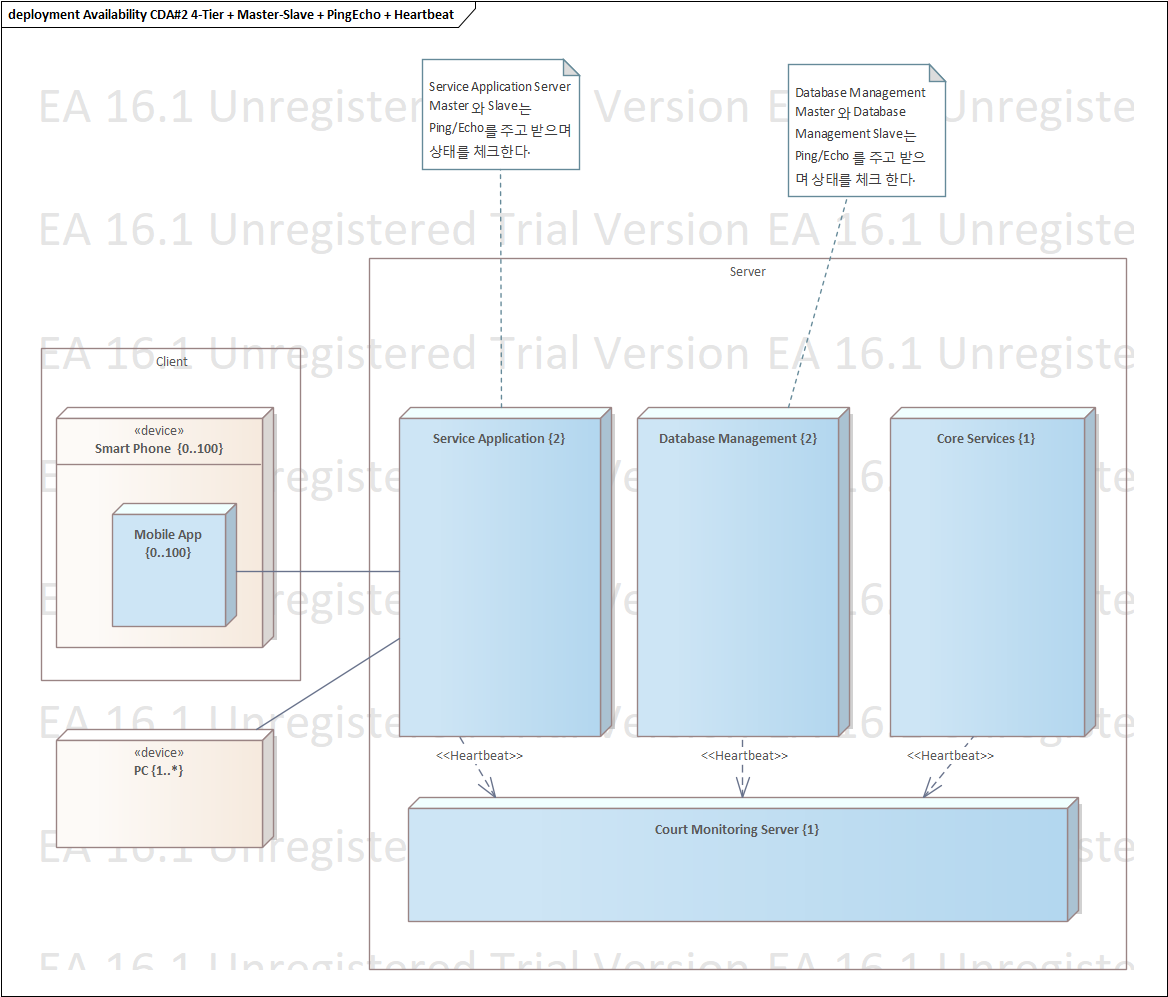
#### CDA #1 Description: Server-Client + Real-time monitoring

|  |  |
| --- | --- |
| QA3\_CD\_CDA-01 | Server-Client + Real-time monitoring |
| **Description** | Client로 동작하게 되는 것은 방청객의 안드로이드 Smartphone에서 동작할 안드로이드 앱 과 소송관계인 및 재판부 PC의 브라우저에서 동작하게 될 웹페이지 이다.  이를 위해서 불필요한 tier를 최소화 한 Service-Client 구조를 채택하고, Real-time monitoring을 할 수 있는 모니터링 컴포넌트를 두어 전체 시스템의 가용성을 탐지 할 수 있게 한다.  ㅁ Service-Client 구조 설명  방청객의 Mobile App 과 재판 관계인들의 PC에서 브라우저로 접속하는 web page를 클라이언트로 두고 이외의 기능들을 서버에서 구현하는 형태이다.  클라이언트 측을 Presentation 계층으로 두어 사용자 인터페이스를 통해 상호 작용할 수 있도록 합니다.  모바일 앱에서는 재판 진행 데이터를 접근하고 시청이 필요한 데이터를 요청할 수 있습니다.  Application/Data/Security Layer를 서버단에 구현하여 서비스를 제공한다.  ㅁ Real-time monitoring 구조 설명   * Service application, Data Management, Core Service Manager, Online Court Service, Broadcasting Service 등 모든 주요 컴포넌트에 agent 설치 * 에이전트는 실시간 시스템 리소스 사용량, 프로세스 상태, 로그 등의 데이터 수집 * 중앙 모니터링 시스템은 수집된 데이터를 저장하고 분석 * 실시간 대시보드 및 알림을 받을수 있음 |
| **Pros** | ㅁ Server-Client 구조 장점   * 디지털 법정 서버측 계층을 수평적으로 확장하여 부하를 분산시 가용성 확보가 용이하다. * 방청객 모바일 앱에 문제가 생겨도 서버가 분리되어 있기 때문에 전체 시스템이 다운되거나 하지 않는다. * 다른 Multi-Tier (3 tier/ 4 tier) 아키텍처 대비 단순하면서 개발/유지보수 측면에서는 비교적 간단하다.   ㅁ Real-time monitoring 장점   * 디지털 법정 시스템의 상태를 실시간으로 파악하여 신속한 대응 가능 * 잠재적 장애 감지 및 예방 가능 * 전체 시스템의 가용성과 성능을 종합 관리 가능 |
| **Cons** | ㅁ Server-Client 구조 단점   * 네트워크 의존성이 있기 때문에 네트워크에 이슈 발생시 방청객이 시스템에 접근하여 정상적인 서비스를 받는데 어려움이 있을 수 있다. * Application/Data/Security 영역이 한 군데 집중되어 있어 하나의 컴포넌트에서 이슈 발생시 다른 영역에 영향을 줄 수도 있다.   ㅁ Real-time monitoring 단점   * 대량의 데이터 처리로 인한 리소스 소모가 큼 * 모니터링 시스템 자체의 가용성과 확장성 보장 필요 * 전문 모니터링 도구와 인력이 필요하여 비용 발생 |

****

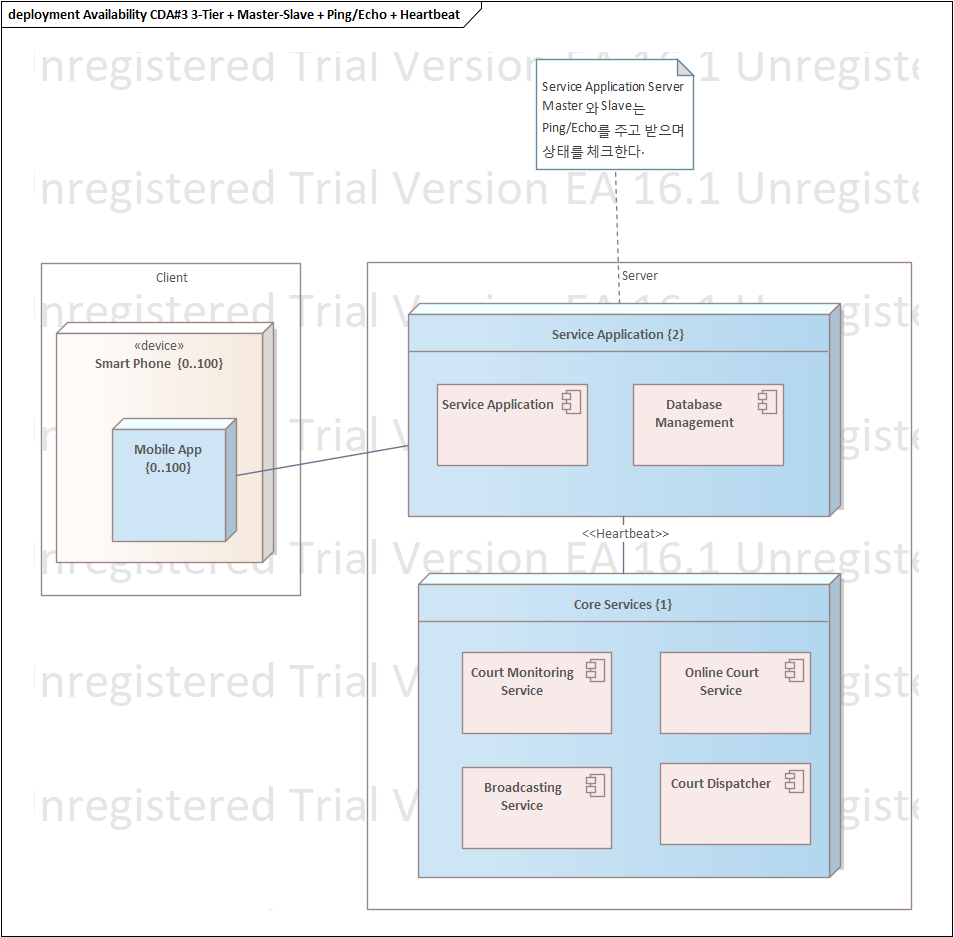
#### CDA #2 Description: Multi-Tier + Master-Slave + Ping/Echo + Heartbeat

|  |  |
| --- | --- |
| QA3\_CD\_CDA-02 | Multi-Tier + Master-Slave + Ping/Echo + Heartbeat |
| **Description** | **Client(1)** / **Service Application Server(2)** / **Database Management Server(3)** / **Core Service Server(4)** / **Court Monitoring Server(5)**로 Multi-Tier를 물리적으로 구분하여, 하나의 컴포넌트에 이슈가 생기더라도 다른 시스템에 문제가 확산 되지 않도록 함  이 때, 주요 컴포넌트는 Master-Slave 방식으로 이중화 시켜 가용성을 향상 시킨다. 뿐만 아니라 주요 서비스는 Heartbeat를 Court Monitoring Server로 주기적으로 보내서 시스템의 health 가 관리될 수 있게 하며, Master-Slave로 이중화 되어 있는 시스템은 ping/echo 방식으로 서로의 health check를 하여 문제 발생시 빠른 복구가 가능하도록 한다. |
| **Pros** | ㅁ높은 가용성   * 물리적 계층 분리로 한 계층의 장애가 전체 시스템에 영향을 미치지 않습니다. * 주요 컴포넌트의 이중화(마스터-슬레이브)로 장애 대비가 가능합니다. * 헬스체크를 통해 문제를 조기에 감지하고 복구할 수 있습니다.   ㅁ 모니터링 용이성   * 중앙 집중식 모니터링 서비스(Court Monitoring Service)를 통해 전체 시스템을 통합 관리할 수 있습니다. |
| **Cons** | 복잡성 증가   * 다중 계층 아키텍처로 인해 전체 시스템의 복잡성이 높아집니다. * 계층 간 통신 및 데이터 흐름 관리가 필요합니다.   운영 오버헤드   * 각 계층과 이중화된 컴포넌트를 개별적으로 관리해야 하는 오버헤드가 있습니다. * 모니터링 시스템 자체에 대한 관리 부담도 있습니다.   초기 구축 비용   * 다중 계층 아키텍처와 이중화, 모니터링 시스템 구축을 위한 초기 투자 비용이 큽니다. |

****

#### CDA #3 Description: 3-Tier + Master-Slave + Ping/Echo + Heartbeat

|  |  |
| --- | --- |
| QA3\_CD\_CDA-03 | 3-Tier + Master-Slave + Ping/Echo + Heartbeat |
| **Description** | **Mobile App Client(1) Service Application Server** 와 **Database Management** **(2)** / **Core Services(3)** - Broadcasting Service, OnlineCourtService, Court Monitoring Service 로 Tier를 구성하고, 주요 컴포넌트는 Master-Slave로 이중화 시켜 가용성을 향상 시킨다. 주요 컴포넌트는 Heartbeat를 Court Monitoring Service로 보내서 시스템의 health 체크가 가능하게 한다.    시스템 운영시 가장 밀접한 관련이 있는 Service Application Server 와 Database Management를 하나의 Tier 아래로 통합하여 운영하는 방식이다.  Tier를 최소화 시켜 Tier 관리에 필요한 오버헤드가 줄어들 것으로 보인다. |
| **Pros** | 가용성 확보   * 마스터-슬레이브 이중화와 헬스체크 시스템을 통해 가용성을 확보할 수 있습니다.   비용 절감   * Multi-tier 대비 계층이 적어 초기 구축 비용과 운영 비용이 절감됩니다. * 서버 계층을 통합하여 하드웨어 리소스 요구량이 줄어듭니다. |
| **Cons** | 보안 고려 사항   * 서버와 데이터베이스 계층이 통합되어 보안 정책 적용이 제한적일 수 있습니다. |

****

#### Decision and Rationale

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Availability* | | **Analysis** | QA3\_CD\_CDA-01 : Server-Client + Real-time monitoring | QA3\_CD\_CDA-02 : Multi-Tier + Master-Slave + Ping/Echo + Heartbeat  **(Selected)** | QA3\_CD\_CDA-03 : 3-Tier + Master-Slave + Ping/Echo + Heartbeat |
| **ID** | **Title** |
| QAS-01 | 오류 없는 원활한 재판 진행 | Pros | (0) 기본적인 Client-Server 구조의 가용성 정도 제공 | (++) 전체 주요 컴포넌트의 독립화 구조로 안정성 확보 | (+) Multi-Tier 대비 비용 감소, Server-Client 구조 보다 높은 가용성 확보 |
| Cons | (--) Data 의 암호화 측면과 서버 계층에 일부 장애 발생시 전체 시스템에 영향을 줄 수 있음 | (-) 하드웨어 계층이 많이 나뉘어 져서 비싸다 | (0) Client-Server 보다는 가용성 측면에서 유리하나 Multi Tier 보다는 떨어짐 |
| QAS-02 | 시스템 장애 탐지 및 복구 | Pros | (++) 실시간 장애 대비 및 확인이 가능하여 시스템 가용성 확보 가능 | (++) 전체 시스템의 health 를 주기적이고 독립적으로 확인 가능함 | (+) 전체 시스템의 health 를 주기적으로 확인 가능함 |
| Cons | (--) Real-Time Monitoring 을 위한 부가 비용 발생 | (-) 모든 컴포넌트가 heartbeat를 전송해야 하는 부담  monitoring system이 이를 다 받아줘야 하는 부담 존재 | (-) 모든 컴포넌트가 heartbeat를 전송해야 하는 부담  monitoring system이 이를 다 받아줘야 하는 부담 존재 |

Candidate Design:

|  |  |  |  |
| --- | --- | --- | --- |
| **QA** | **QAS** | **CD** | **Description** |
| QA1:  *Availability* | QAS-04,  QAS-05 | Multi-Tier + Master-Slave + Ping/Echo + Heartbeat | 모듈별로 동작하는 서버의 Tier를 명확하게 나누고 각 서버의 Health Check 진행함에 따라 높은 가용성을 확보 할 수 있다. |

### QA4: Usability

#### Design Goal

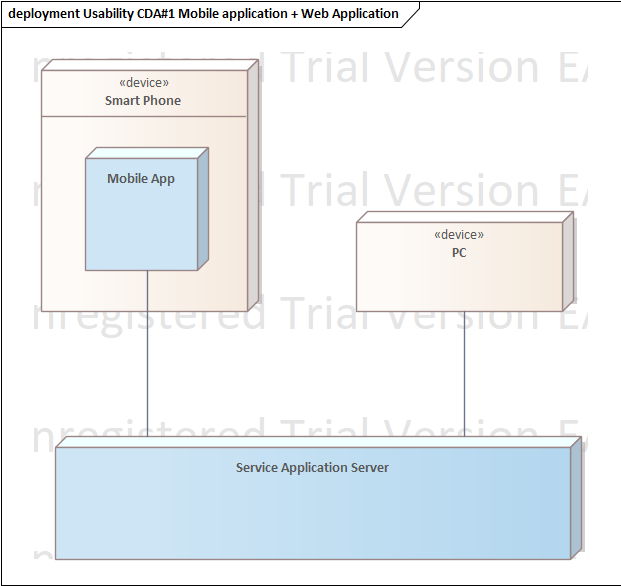
시스템은 여러 가지 면에서 사용자가 편리하고 효율적으로 기능을 사용할 수 있게 해야 한다. 그 중에서 우리는 시스템의 효율성을 높이고, 추후 사용자 편의성의 관심이 변할 때 빠르게 이를 대응할 수 있는 Architecture를 적용하고자 한다. 우리 시스템에서 사용자와 직접적인 interaction이 이루어지는 부분은 방청객이 사용하는 Mobile App과 재판 중에 시스템을 직접 이용하게 될 재판부, 소송 관계인의 Browser이다. 관련된 **Reference Architecture** 를 찾아보면 **Mobile Application**과 **Web Application**이 기본적으로 존재하고, 추가로 Mobile App에 **Interaction Oriented Architecture** 중 **PAC, MVC**의 적용을 확인 해 본다. 마지막으로 **Rich Internet Application** 을 추가로 확인하여 Usability 향상에 도움되는 요소가 있는지 검토해 보기로 한다.

다음은 Usability 측면에서의 3가지 Candidate Design Approach이다.

#### Candidate Design Approach List

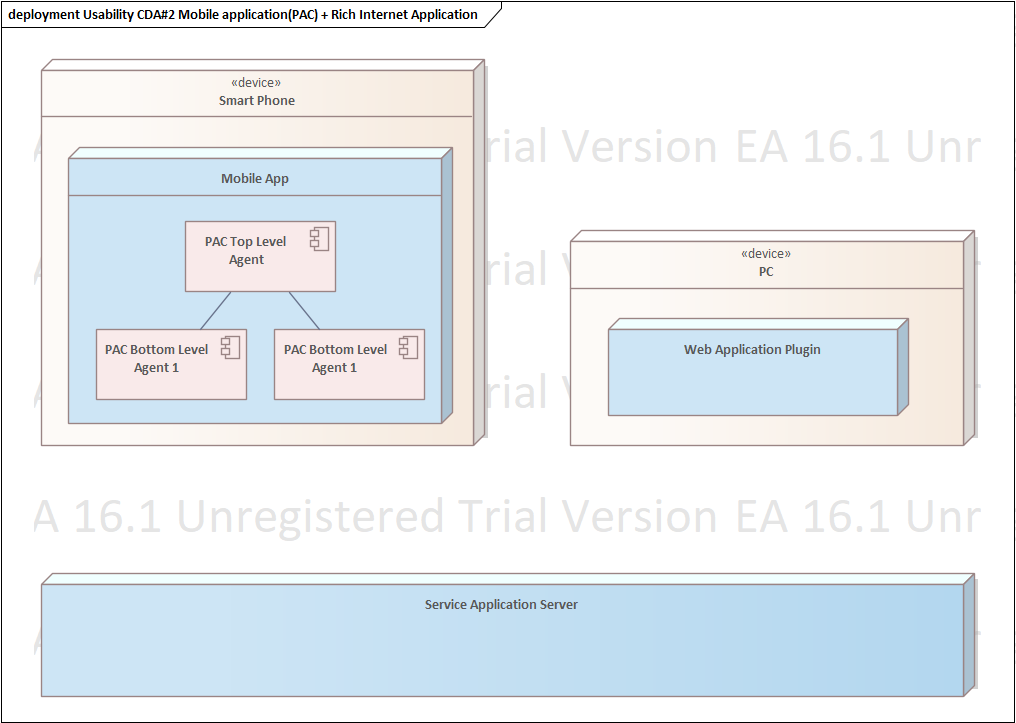
#### CDA #1 Description: Mobile application + Web Application

|  |  |
| --- | --- |
| QA4\_CD\_CDA-01 | Mobile application + Web Application |
| **Description** | 디지털 법정시스템에서 사용자와 직접 인터렉션이 이루어 지는 곳은 방청객의 모바일 앱과 소송관계자 PC, 재판부 PC 등이 있을 수 있다.  이 상황에서 사용성을 확보하여 사용자가 효율적으로 기능을 사용할 수 있도록 기능을 제공하기 위해서 각자의 상황에서 접근성이 용이한 Mobile 과 Web application 구조를 채택한다.  이를 통해 스마트폰을 사용하는 방청객은 디지털 법정과의 연동성이 최적화된 앱을 사용할 수 있고, 소송관계자와 재판부는 법정 내에서 사용가능한 PC에서 쉽게 시스템을 접근하여 필요한 작업을 할 수 있게 한다. |
| **Pros** | 방청객의 모바일 앱: 모바일 기기에 최적화된 사용자 경험 제공  소송관계자 PC, 재판부 PC - 웹 앱: 크로스 플랫폼 호환성으로 다양한 기기 및 브라우저에서 접근 가능 |
| **Cons** | 모바일 앱: 특정 플랫폼에 종속적이며, 앱 스토어의 승인 필요, 향후 다른 모바일 플랫폼으로 확장시 신규 개발 필요  웹 앱: 오프라인 사용에 제한적이며, 모바일 기기의 하드웨어 기능에 접근이 제한될 수 있음 |

****

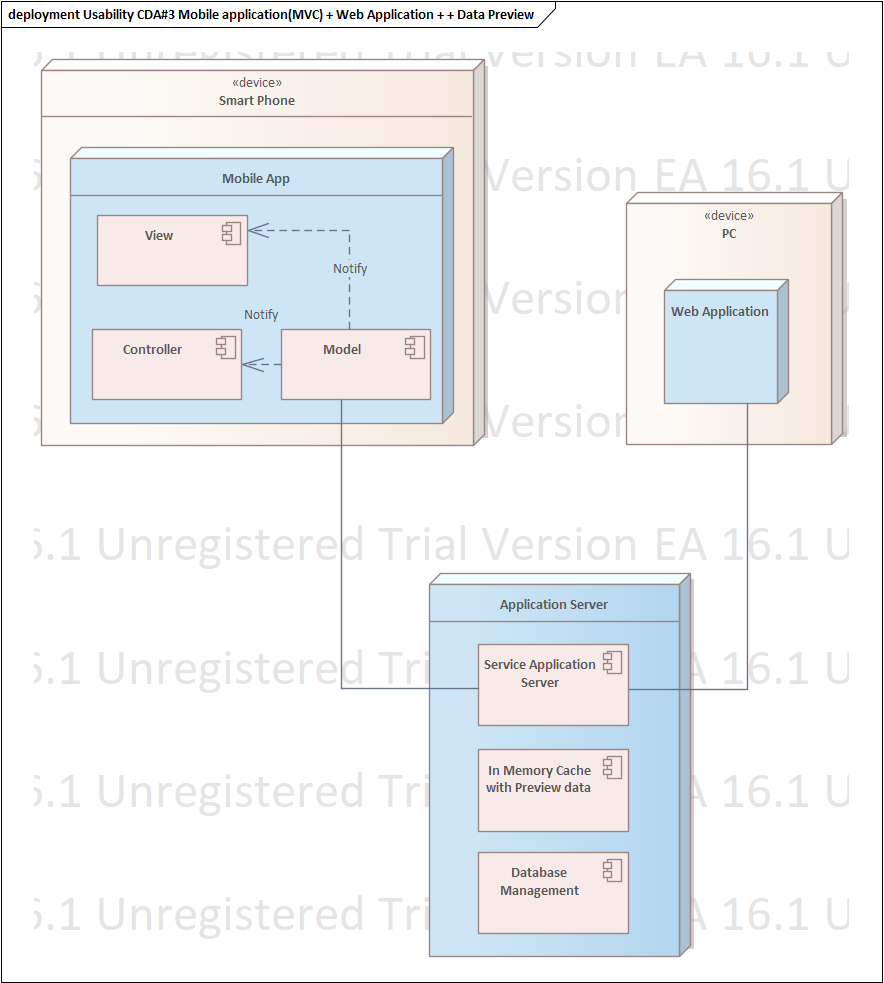
#### CDA #2 Description: Mobile application(PAC) + Rich Internet Application

|  |  |
| --- | --- |
| QA4\_CD\_CDA-02 | Mobile application(PAC) + Rich Internet Application |
| **Description** | Mobile App 개발시 PAC (Presentation Abstraction Control) 패턴을 적용하여 사용성과 관련된 기능 (4회 이하 클릭으로 자료입력, seamless 한 재판 자료 보기 등)을 개발자가 구조적으로 쉽게 개발 할 수 있는 방법을 제공한다.  뿐만 아니라, 입력된 자료의 제어를 그래픽으로 할 때 Browser 기본 기능 외에 확장 Plugin을 쓸 수 있는 Rich Internet Application 구조를 적용하여, 재판부가 공용 스크린으로 자료 제어를 하여 보여주기를 진행할 때 직관적인 그래픽을 사용할 수 있게 한다. |
| **Pros** | ㅁ Mobile application(PAC) 장점   * 모듈화된 구조로 유연하고 확장가능 * 사용자 인터페이스와 비즈니스 로직을 분리하여 유지보수 용이   ㅁ Rich Internet Application 장점   * 풍부한 그래픽 및 인터랙션으로 사용자 경험을 향상 시킬 수 있음 * 일부기능 오프라인 기능 제공이 가능 * 클라이언트 측 로직 실행으로 서버 요청이 줄어들 수 있는 경우 성능 개선 가능 (브라우저의 캐시 기능 보다 더 복잡한 기능이 필요한 경우) |
| **Cons** | ㅁ Mobile application(PAC) 단점   * 방청객이 사용할 앱의 기능을 고려할 때 비교적 UI의 계층을 나눌 만큼 복잡하지 않아 복잡도 대비 장점이 축소 될 수 있음 * 복잡성이 증가할 수 있고, 초기 설계가 어려울 수 있음   ㅁ Rich Internet Application 단점   * 브라우저 사용 외에 플러그인 설치가 필요하다 * 다양한 브라우저 및 플랫폼에서의 호환성 유지가 도전적일수 있음 * 클라이언트 측 실행으로 인한 보안 취약점 발생 가능 |

****

#### CDA #3 Description: Mobile application(MVC) + Web Application + + Data Preview

|  |  |
| --- | --- |
| QA4\_CD\_CDA-03 | Mobile application(MVC) + Web Application + Data Preview |
| **Description** | 방청객이 사용하는 모바일 앱 구현시 MVC 패턴을 적용하여 인터랙션의 확장과 변경에 유연한 구조를 확보하며 (4회 이하 클릭으로 자료입력, seamless 한 재판 자료 보기 등) 개발자가 구조적으로 쉽게 사용성과 관련된 개발을 할 수 있는 방법을 제공한다.  또한 서버 data 변경시 observer 패턴을 통해 변경 상태를 확인하고, 변경된 data를 앱의 상태에 반영할 수 있게 하여 보다 효율적인 사용을 할 수 있게 한다.  ㅁ Web Application은 CDA#1과 동일  ㅁ Data Preview는 In Memory Cache 컴포넌트에서 Data를 빠른 메모리로 올려둘 때, full data 뿐만 아니라 preview 형태의 적은 용량의 data로 정보를 preview 할 수 있는 형태로 따로 저장해두고, 이를 클라이언트가 보다 빠르게 확인할 수 있게 하여 사용자의 seamless 한 사용자 경험에 기여한다. |
| **Pros** | ㅁ MVC 장점   * 표현 부분과 데이터 저장 부분을 분리 시켜 이후 사용성 관련 UI 개발이 전체 앱 기능과 독립적으로 빠르게 개발 및 유지보수 가능하게 함 * 변경의 전달이 consistency 하게 이루어 질 수 있게 진행 * View 나 Controller가 실시간 pluggable 하다.   ㅁ Data Preview 장점   * 방청객이 재판 자료를 보려할 때 항상 full data를 download 받게 하는 것은 download 시간이 오래걸려 seamless 한 사용자 경험을 방해할 수 있다. 용량이 큰 데이터의 Preview를 제공함으로써, Preview를 보는 동안 background에서 full data를 download 할 수 있게 하여 사용자 경험을 향상 시킬 수 있게 한다. |
| **Cons** | ㅁ MVC 단점   * MVC 패턴을 모르는 경우 초기 개발이 오래걸릴 수 있다.   ㅁ Data Preview 단점   * Preview 생성 로직 개발 필요 * Preview data의 보관이 부가적으로 필요하여 비용 발생 |

****

#### Decision and Rationale

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Usability | | **Analysis** | QA4\_CD\_CDA-01 : Mobile application + Web Application | QA4\_CD\_CDA-02 : Mobile application(PAC) + Rich Internet Application | QA4\_CD\_CDA-03 : Mobile application(MVC) + Web Application + Data Preview  **(Selected)** |
| **ID** | **Title** |
| QAS-06 | 편리한 재판 자료 제어 | Pros | (0) | (+) PAC 적용으로 인한 독립적인 UI 가능한 구조, Plug In 제공을 통한 브라우저 내에서 Graphic 한 Control 가능 | (++) MVC 적용으로 인한 독립적인 UI 가능한 구조 확보, Data Preview 기능 제공으로 seamless한 사용자 경험 제공 가능한 구조 확보 |
| Cons | (0) | (-) PAC 개발이 어려움, Plug in 설치 및 브라우저 호환성 확보, 보안 처리등이 어려움 | (-) MVC 개발이 어려움, Data Preview 기능 로직 개발이 복잡함 |

Candidate Design:

|  |  |  |  |
| --- | --- | --- | --- |
| **QA** | **QAS** | **CD** | **Description** |
| QA1:  Usability | QAS-06 | QA4\_CD\_CDA-03 : Mobile application(MVC) + Web Application + Data Preview | 방청객 앱의 기능이 계층적 구조를 필요로 하지는 않다는 점과 사용성 개발의 용이성을 고려하여 Mobile Application(MVC)을 채택하고 plug in 제공시 단점 중 호환성 확보와 보완처리의 어려움을 고려하여 Web Application 장식을 채택한다. 마지막으로 Data Preview의 경우 부가적인 사용성을 확보 할 수 있다는 점을 고려하여 보다 나은 사용성 확보를 위해 채택하기로 한다. |

### QA5: Security

#### Design Goal

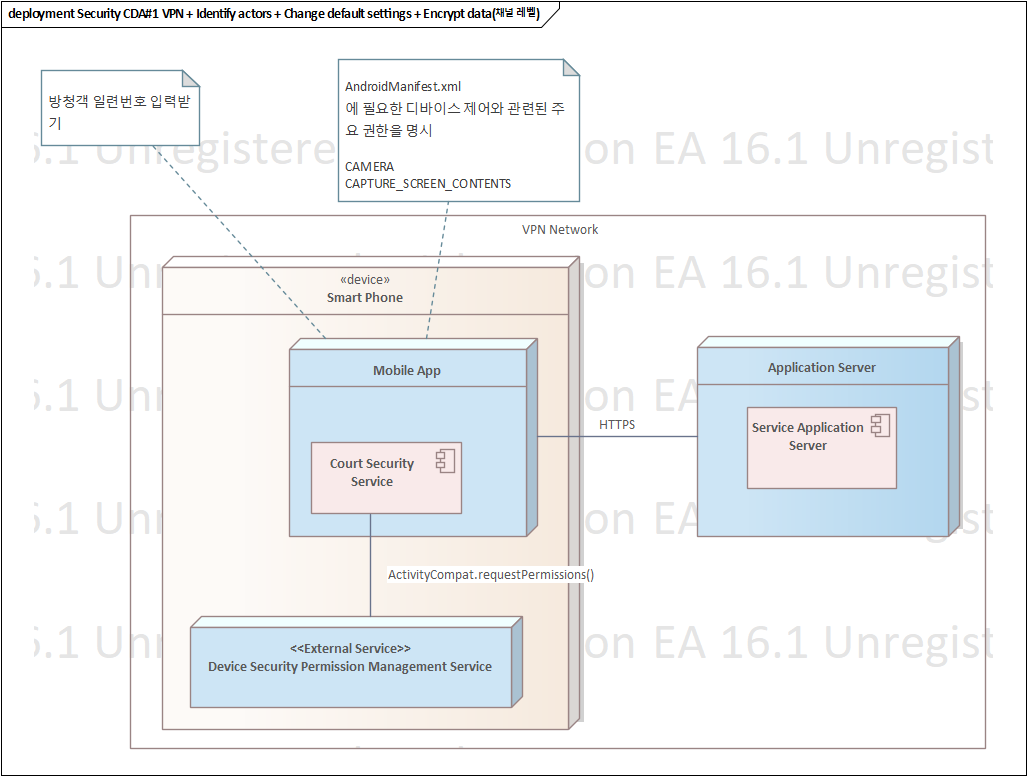
법원은 본 시스템이 정상 동작될 때, 재판 정보가 임의로 외부 노출되지 않길 원합니다. 이를 위해 반드시 필요한 사람만 본 시스템에 접근 가능하며, 접근을 한 사용자들 중에서도 사용 당사자가 사용해야하는 기준에 맞는 동작만을 쓸 수 있도록 제어 할 수 있길 바랍니다. 이를 위해 기본적인 접근제어를 위해 **VPN** 사용, Architectural Tactic 중 Security 항목 들인 **Identify actors** 와 **Change default settings의 적용 유무** 그리고 **채널/콘텐츠 레벨의 Data Encryption 적용 여부**를 검토하여 Security를 만족시킬 수 있는 방법을 찾고자 합니다.

다음은 Security 측면에서의 3가지 Candidate Design Approach 이다.

#### Candidate Design Approach List

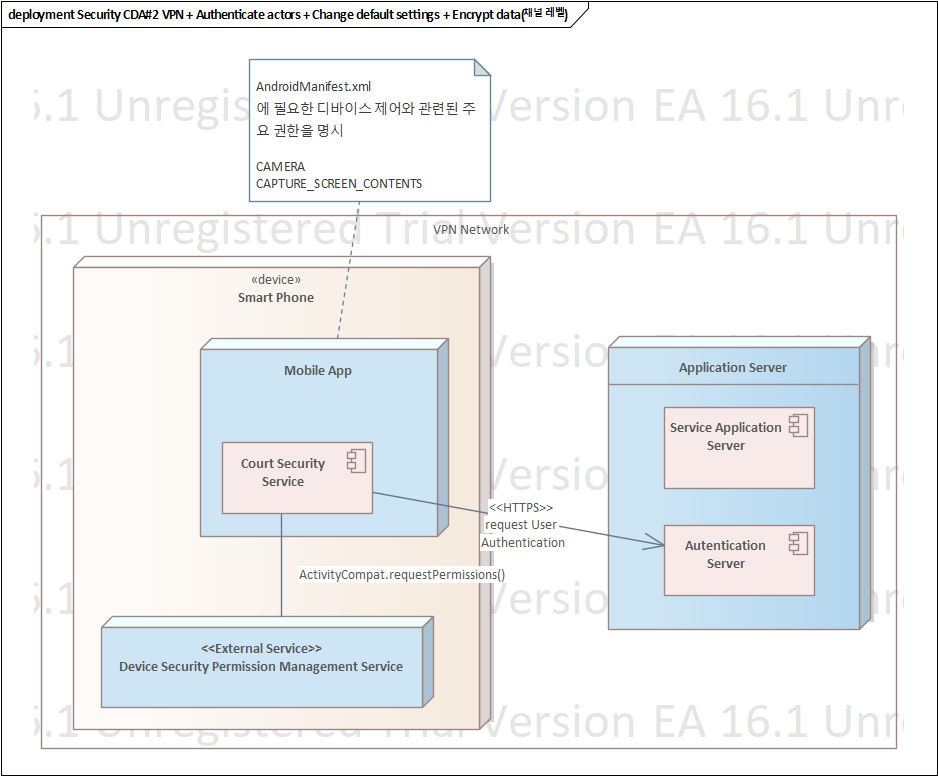
#### CDA #1 Description: VPN + Identify actors + Change default settings + Encrypt data(채널 레벨)

|  |  |
| --- | --- |
| QA5\_CD\_CDA-01 | VPN + Identify actors + Change default settings + Encrypt data(채널 레벨) |
| **Description** | VPN 을 구성하여 디지털 법정 시스템을 그 안에서 동작하게 함으로써, VPN 외부에서의 접근을 제한하고 불필요한 정보의 노출을 막는다.  방청객의 경우 디지털 법정 앱은 Play Store에서 다운로드 받을 수 있지만, 실제 시스템에 접근을 위해서는 법정에서 제공하는 VPN 네트워크에 접근 해야만 하고 이는 [**Assumption에 이야기한 것 처럼**](#_heading=h.3rdcrjn) **법원에서 제공하는 법정 앞에서 VPN 네트워크 망의 패스워드를 확인하고 스마트폰을 연결** 함으로써 가능하다.  Identify actors   * 시스템은 방청객이 방청 신청시 발급 받는 일련번호를 입력하여 재판의 방청객임을 식별 받게 한다. 이 정보는 재판의 기준정보 설정시 시스템 관리자가 해당 재판 정보로 세팅하게 되며, 이후 Security Manager를 통해서 사용자를 식별하게 된다. * 소송관련자 PC와 재판부 PC는 법정 내에 존재하는 장치로 이들은 VPN내에 존재하여 장치별 사용 레벨을 식별할 수 있게 한다. 이 역시 Security Manager를 통해서 기능 사용 레벨이 관리 된다.   Change default settings   * [방청객(안드로이드 스마트폰)](#_heading=h.3rdcrjn) 은 사용자 식별 후 재판 시작후 부터 재판부의 승인을 받은 데이터를 접근 할 수 있는데, 이 때 사용자 **앱의 보안 정책**이 실행 되며 이를 통해 **모바일은 카메라 기능, 화면 녹화 및 캡쳐 기능 이 잠기게 됩니다**. 해당 세팅은 사용자가 VPN 에서 벗어나면 해지 되지만, 그와 동시에 앱에서 접근하던 재판 데이터도 더이상 확인할 수 없는 상태가 됩니다. * 기본적으로 방청객 앱을 통해 확인한 재판 데이터는 저장하지 않으며 이를 통해 불필요한 정보 노출을 막습니다.   Encrypt data(채널 레벨) 적용시 전체 데이터 스트림을 암호화 하는 반면, 컨텐츠 레벨 처럼 개별 데이터를 암호화 하지는 않음, 여기서 말하는 전체 시스템은 클라이언트와 Service Application Server를 말하며, 외부 시스템과의 영상 데이터 처리 부분을 말 하진 않는다. |
| **Pros** | VPN 장점   * 기밀성과 무결성 보장, 민감한 정보 유출 위험 최소화   Identify actors 장점   * actor를 식별하여 actor 외 다른 사람이 쓰는 것을 방지   Change default settings 장점   * 사용자의 스마트폰 장치의 기본 설정을 변경하여 정보 노출을 방지   Encrypt data(채널 레벨) 장점   * 비교적 간단하게 구현이 가능 * 채널 수전에서 전체 데이터 스트림을 보호함 |
| **Cons** | VPN 단점   * 복잡한 구축 및 관리(VPN 서버 구축, 인증서 관리, 정책 설정 등 복잡한 구성 작업이 필요함)   Identify actors 단점   * Actor 식별로 사용되는 일련번호가 노출되는 위험이 있음   Change default settings 단점   * 2 대 이상의 디지털 장비가 있을 시 촬영으로 회피 가능   Encrypt data(채널 레벨) 단점   * 데이터의 송수신 지점(endpoint)인 방청객 모바일 앱 도달 이후는 암호화되지 않습니다. |

****

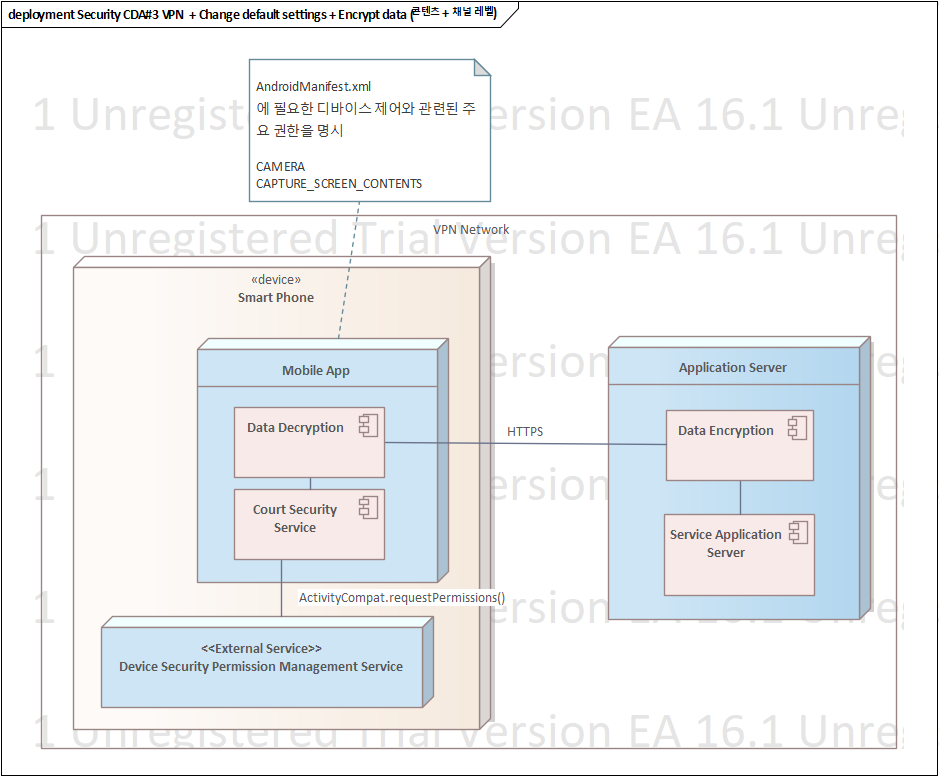
#### CDA #2 Description: VPN + Authenticate actors + Change default settings + Encrypt data(채널 레벨)

|  |  |
| --- | --- |
| QA5\_CD\_CDA-02 | VPN + Authenticate actors + Change default settings + Encrypt data(채널 레벨) |
| **Description** | VPN, Change default settings, Encrypt data(채널 레벨) 은 CDA#1과 동일  Authenticate actors 방식을 적용하여 실제 사용자가 동일인인지 파악하는 단계가 부가적으로 들어가는 것을 말하며, 부가적인 사용자 이름과 비밀번호, 생체인식등의 방식을 통해 이루어 집니다. |
| **Pros** | Authenticate actors 방식 장점   * 식별 방법보다 보안 수준이 높음 |
| **Cons** | Authenticate actors 방식 단점   * 관리 오버헤드가 있음, 본시스템이 가지고 있는 정보 외에 비밀번호 관리 등이 부가적으로 필요함 |

****

#### CDA #3 Description: VPN + Change default settings + Encrypt data (콘텐츠 + 채널 레벨)

|  |  |
| --- | --- |
| QA5\_CD\_CDA-03 | VPN + Change default settings + Encrypt data(콘텐츠 + 채널 레벨) |
| **Description** | VPN와 Change default settings은 CDA#1 과 동일  재판에 입장하는 것 자체가 방청객 신청을 완료한 사람이 아니면 불가능하다는 점을 고려하여, 사용자 인증 방식 자체를 생략한다.  Encrypt data(콘텐츠 + 채널 레벨) 적용시 시스템 내 채널의 데이터 스트림 암호화 뿐만 아니라 실제 전달 되는 콘텐츠의 암호화도 같이 적용하는 것을 말한다. |
| **Pros** | VPN와 Change default settings은 CDA#1 과 동일  ㅁ Encrypt data(콘텐츠 + 채널 레벨) 적용 장점   * 시스템 채널과 콘텐츠가 암호화 됨에 따라 데이터 자체가 보호되므로 보안 수준이 높음 * 콘텐츠가 외부로 노출 되더라도, 방청객이 사용한 앱을 통해 decrypt 되지 않으면 사용 불가하게 하여 보안 유지 |
| **Cons** | VPN와 Change default settings은 CDA#1 과 동일  ㅁ Encrypt data(콘텐츠 + 채널 레벨) 적용 단점   * 구현이 복잡함 * 채널 암호화 뿐만 아니라 콘텐츠 암호화 적용으로 인해 암호화 알고리즘을 양쪽에 적용 후 콘텐츠 보관 이 필요하며 인코딩/디코딩이 필요해 성능 저하가 발생할 수 있음 |



#### Decision and Rationale

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Security* | | **Analysis** | QA5\_CD\_CDA-01 : VPN + Identify actors + Change default settings + Encrypt data(채널 레벨)  **(Selected)** | QA5\_CD\_CDA-02 : VPN + Authenticate actors + Change default settings + Encrypt data(채널 레벨) | QA5\_CD\_CDA-03 : VPN + Change default settings + Encrypt data(콘텐츠 + 채널 레벨) |
| **ID** | **Title** |
| QAS-07 | 재판 자료 접근 제어 | Pros | (++) 사용자를 명확하게 확인 하고 이를 통한 접근 제어 가능, 불필요한 데이터 노출 제한 가능  시스템 진입점으로 사용자 구분하여 접근 제어 가능 | (++) 사용자 확인 후 실제 사용자를 한번 더 인증 하는 방식으로 시스템 사용자의 신원을 명확히 함  시스템 진입점으로 사용자 구분하여 접근 제어 가능 | (++) 데이터 암호화를 강력하게 적용하여 정보 누출 방지함,  시스템 진입점으로 사용자 구분하여 접근 제어 가능 |
| Cons | (-) VPN 개발이 필요하여 시스템 구축과 운영이 상대적으로 복잡해질 수 있음 | (--) 사용자 정보를 추가적으로 확인하는 방법 필요 | (--) 컨텐츠 암호화 적용으로 성능 저하 가능성 있음, 암호화 적용 알고리즘 개발이 복잡함 |

Candidate Design:

|  |  |  |  |
| --- | --- | --- | --- |
| **QA** | **QAS** | **CD** | **Description** |
| QA1:  Security | QAS-07 | QA5\_CD\_CDA-01 : VPN + Identify actors + Change default settings + Encrypt data(채널 레벨) | Security 측면에서 고려시, VPN으로 네트워크를 일반 인터넷과 분리시켰기 때문에 기본적인 안정성 확보는 했다고 고려하여, 꼭 필요한 부분인 사용자 확인, 모바일 앱의 보안 설정과 채널 암호화를 채택 하였다.  이 방안보다 더 높은 보안 적용시 부가적인 정보를 가지고 있어야 하는 부분과 성능에 영향을 주는 부분 등 부정적인 영향을 주는 항목이 더 커져서 1번 안을 채택 한다. |

## 

## Candidate Designs Evaluation for all QAs

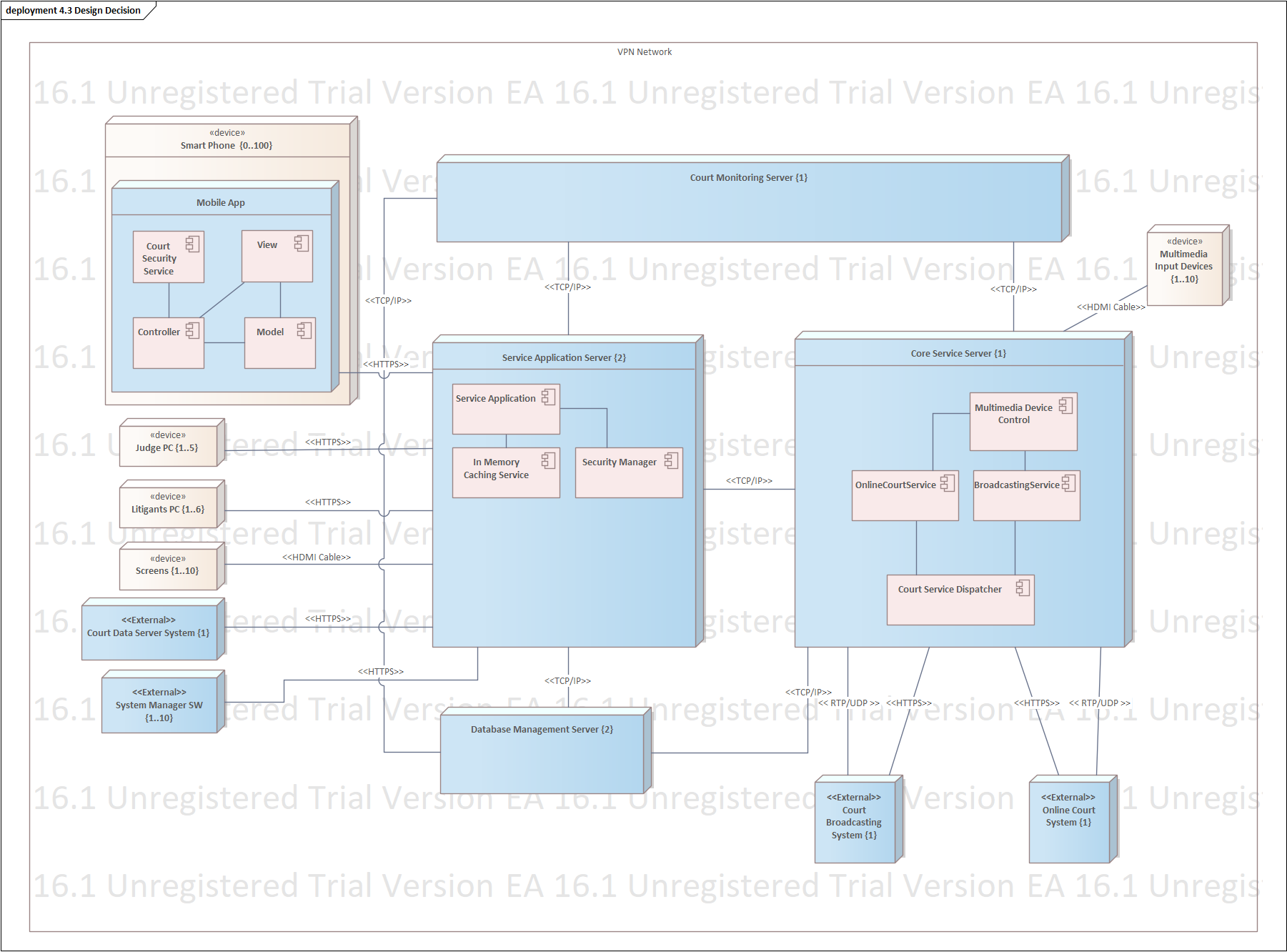
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **QA** | **QAS** | **Analysis** | **Candidate Design (CD) #1**  QA1\_CD\_CDA-03 : Web RTC 방식 + DB Master-Slave + in memory caching | **Candidate Design (CD) #2**  QA2\_CD\_CDA-03 : Dispatcher | **Candidate Design (CD) #3**  QA3\_CD\_CDA-02 : Multi-Tier + Master-Slave + Ping/Echo + Heartbeat | **Candidate Design (CD) #4**  QA4\_CD\_CDA-03 : Mobile application(MVC) + Web Application + Data Preview | **Candidate Design (CD) #5**  QA5\_CD\_CDA-01 : VPN + Identify actors + Change default settings + Encrypt data(채널 레벨) |
| **QA1**  **Performance** | QAS-01  QAS-02 | Pros | (++) Web RTC 방식으로 영상 법정 연동시 Low Latency 보장, In Memory Caching을 통한 빠른 성능 확보 | (NA) | (+)  Master-Slave 방식 적용으로 시스템 오류 상태에서 빠른 복구 가능 | (+) Data Preview로 사용자에게 빠른 데이터 접근성 제공 | (NA) |
| Cons | (-) Caching 컴포넌트를 위한 비용 증가 | (-) Dispatcher를 통해야만 시스템 연동 가능 | (-) 서버 내부가 Tier로 나뉘어서 Tier간 통신으로 성능이 늦어질 수 있음 | (NA) | (-) 채널 암호화 적용으로 인한 부가적인 CPU /메모리 사용량 발생 |
| **QA2**  **Extensibility** | QAS-03 | Pros | (NA) | (++) 모듈화된 서비스 개발로 향후 서비스 확장시 좋음 | (+) Tier로 나뉘어져 있어 기능 확장에 유리함 | (+) 클라이언트/서버 각각 기능 확장에 열려있으며 의존성 적음 | (NA) |
| Cons | (NA) | (NA) | (NA) | (NA) | (NA) |
| **QA3**  **Availability** | QAS-04  QAS-05 | Pros | (+) DB의 Master-Slave 화로 인한 가용성 향상 | (+) Fault tolerance가 일부 가능 | (++) Tier로 나뉘어 각 티어에 존재하는 기능에 문제가 발생하여도 전체 시스템에 영향은 최소화 됨 | (+) 서버 클라이언트 시스템에 문제 발생하여도 다른 곳으로 이슈 확산되지 않음 | (NA) |
| Cons | (-) Master-Slave 관리 복잡성 증가 | (-) Dispatcher의 Single Point of Failure 발생 가능 | (-) Tier가 생기는데 따른 장비 사용으로 비용 증가 | (NA) | (NA) |
| **QA4**  **Usability** | QAS-06 | Pros | (+) 빠른 사용성 | (NA) | (NA) | (++) MVC 패턴 적용으로 향후 변경이 잦은 UI 사용성 이슈 대응에 유리한 구조 확보 | (-) 보안 기능 적용으로 인한 사용자 만족도 저하 있을 수 있음 |
| Cons | (NA) | (NA) | (NA) | (NA) | (NA) |
| **QA5**  **Security** | QAS-07 | Pros | (NA) | (+) 보안기능 집중화 가능 | (+) Tier 별 보안 적용 가능 | (+) 서버 클라이언트 별도의 security 적용가능 | (++) 보안 기능 적용을 통한 불필요한 접근 제어 및 법정 관련 데이터 노출 방지 |
| Cons | (NA) | (NA) | (NA) | (NA) | (NA) |

## Design Decision

본시스템은 VPN으로 구성된 네트워크 내에서 동작되며, 서로 다른 환경에서의 클라이언트 API 호출을 지원하기 위해 HTTPS 프로토콜로 통신한다. 서버 시스템에 접근 할 수 있는 Max Clients(Mobile App 100, PC 30) 수가 제한적이기 때문에 따로 load balancer는 두지 않는다. 대신 Service Application Server 내에 In Memory Caching 컴포넌트를 두어, 진행 중인 재판의 멀티미디어 데이터를 캐싱해 두어 가용성을 높인다. Service Application Server는 TCP/IP통신을 통해 Database Management 서버와 통신하여 데이터 관련 처리를 진행하며, 외부 시스템과의 연동은 향후 확장성을 고려하여 Dispatcher 를 통해 연동 서버 정보가 교환 되게 한다. 이들은 Dispatcher 와 외부 시스템 간의 통신은 HTTPS 방식을 사용하며, 외부 시스템과 영상 정보는 RTP/UDP 프로토콜로 통신한다.

다음은 본 시스템에 존재하는 노드에 대한 간략한 설명이다.

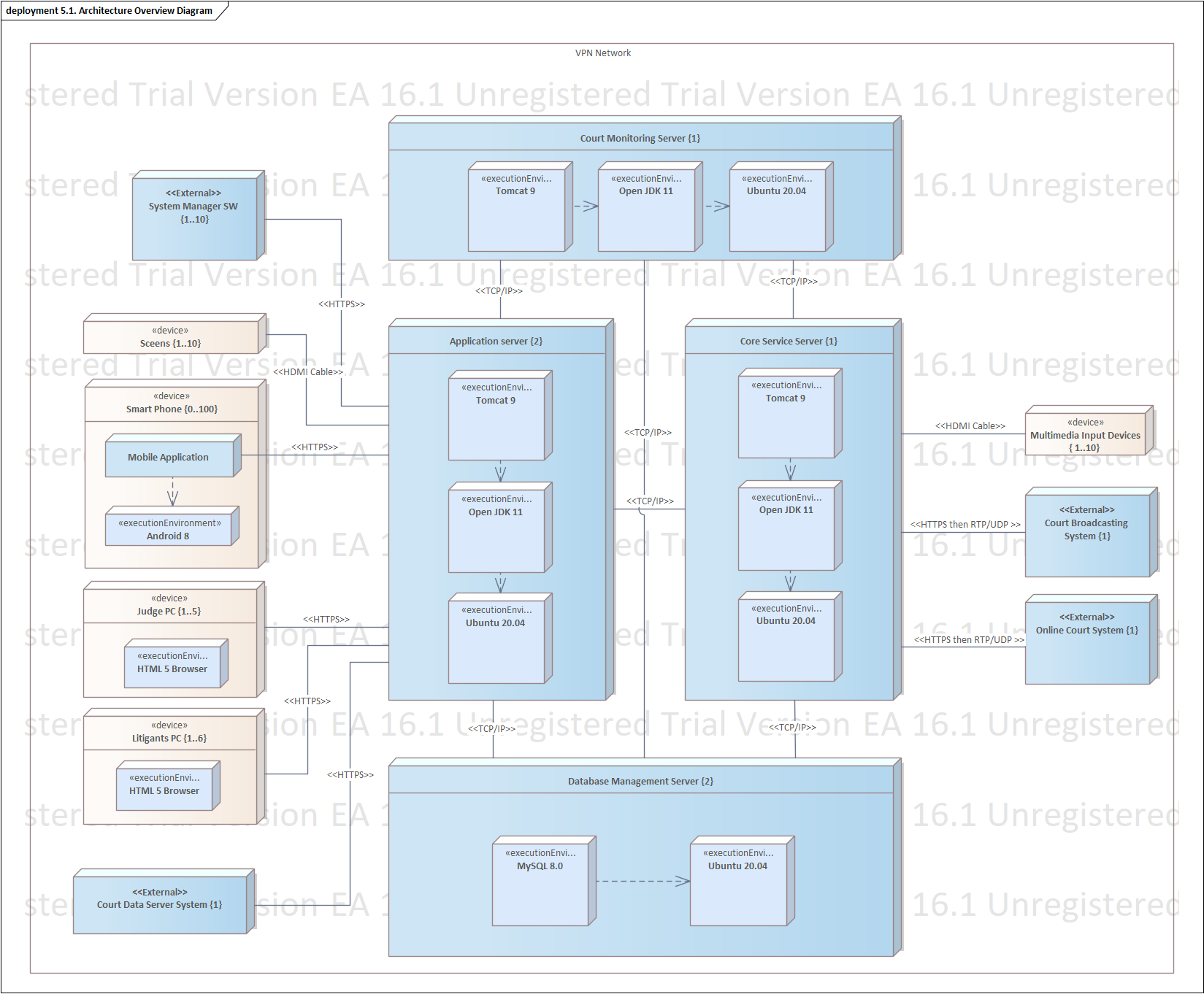
1. VPN Network - 디지털 법정은 시스템이 동작되는 네트워크로 아래의 노드들은 해당 VPN Network 내에서만 동작된다.
2. Mobile App - 방청객의 Smartphone에 설치되어 본 시스템의 Service Application Server와 연동하여 재판 멀티미디어 데이터를 볼 수 있게 한다. 방청객 일련번호 입력 후 재판 정보 조회가 가능하며, 조회가 시작 되기 전에 앱을 통해 스마트폰의 법원 보안설정이 동작된다.
3. Service Application Server - Mobile App 앱과 연동하여 필요한 API를 제공하며, 또한 Court 내에 존재하는 PC에서 Browser로 접속하여 시스템의 기능을 사용할 수 있게 한다. Service Application Server는 Master-Slave 방식으로 2대를 배치하고, Database 와의 통신을 통해 가져온 진행중인 재판의 자료는 In Memory Cache로 가지고 있어 가용성과 성능을 높인다.
4. Database Management Server - 재판의 전/후에 관리자 SW를 통해 법원 자료 서버로 부터 재판의 기준 정보 등을 sync 받는다. Master-Slave 방식으로 2대를 배치하여 가용성을 높인다.
5. Core Service Server - 영상 재판 / 법원 TV 등 외부 시스템과의 연동을 지원하며 멀티미디어 장비로 부터 들어오는 여러 영상 소스를 인코딩 하여 외부로 전달 될 수 있게 지원 한다.
6. Court Monitoring Server - 디지털 법정의 전반적인 운영상황을 주기적으로 모니터링 하고, 필요한 내용을 확인 할 수 있다.



# Architecture Design Description

## Architecture Overview

### Architecture Overview Diagram



### Node Specification

|  |  |
| --- | --- |
| **Name** | **Description** |
| Smart Phone | 역할: 방청객 스마트폰  Multiplicity: 0 .. 100   * 재판 별 100명 이하이며 인당 1대의 장치만 허용   특성   * 디지털 법정 모바일 앱을 실행시키기 위한 스마트폰 * Android OS 버전 8 이상이며 네트워크 연결이 가능한 기기 |
| Judge PC | 재판부가 사용할 PC  Multiplicity: 1..5  특성   * PC는 인터넷 연결이 가능한 듀얼 코어 / 16GB 이상 * 브라우저로 시스템에 접속하여 재판 관련 자료 전송 및 확인 조작이 가능하다. |
| Litigants PC | 역할: 소송 관계자가 사용할 PC  소송관계인 1..6,  특성   * PC는 인터넷 연결이 가능한 듀얼 코어 / 16GB 이상 * 브라우저로 시스템에 접속하여 재판 관련 자료 전송 및 확인 조작이 가능하다. |
| Screens | 역할: 본 시스템의 제어를 받아 승인 받은 멀티미디어 자료를 보여주는 장치  Multiplicity: 1..10  특성  - 공용 스크린: 재판부에서 승인한 자료를 화면에 보여줌  - 영상 법정 전용 스크린: 영상법정 시스템에서 들어오는 영상을 보여주는 장치  - 빔 프로젝터: 해상도 4K UHD (3840x2160) , 최대 화면 140인치, 밝기 2200 안시 이상  - TV: 해상도 4K UHD (3840x2160) , 70 인치 |
| Multimedia Input Devices | 역할: 본 시스템의 제어를 받아 멀티미디어 영상 소스를 입력 시키는  장치  - 카메라: 촬영된 영상을 본 시스템으로 전송  - 실물 화상기: 실물 입체 영상을 본 시스템을 통해 공용 스크린으로 전달  Multiplicity: 1..10  특성  - 카메라: FHD 해상도 60 fps  - 실물 화상기: 500만 화소, 16배 줌 |
| VPN | Virtual Private Network 으로 디지털 법정 시스템이 속한 네트워크이다. |
| Court Data Server System | 역할: 법원 자료 서버로 본 시스템의 자료 조회/입력 요청을 받아 결과를 전달하는 시스템  Multiplicity: 1  특성   * HTTP 서비스가 가능한 외부 서버 |
| Court Broadcasting System | 역할: 법원 TV 시스템   * 본 서비스와 연동되어 재판을 생중계 하는 서비스를 한다.   Multiplicity: 1  특성   * 재판 내용을 중계 하는 외부 시스템 * HTTPS 서비스가 가능한 외부 서버 |
| Online Court System | 역할: 영상 법정 시스템   * 본 서비스와 연동되어 온라인 영상 법정을 운영하는 서비스를 한다.   Multiplicity: 1  특성   * 온라인 영상 법정을 운영하는 외부 시스템 * HTTPS 서비스가 가능한 외부 서버 |
| System Manager SW | 역할: 시스템 관리자 SW   * 본 서비스와 연동되어 재판이 시작 전 후에 법원 자료 서버 를 통해 재판의 기준 정보를 가져와서 디지털 법정 시스템을 세팅 하며, 모니터링 시스템과 연동하여 본 시스템의 안정적인 운영을 확인 한다.   Multiplicity: 1..10  특성   * HTTPS 서비스가 가능한 SW |
| Mobile Application | 역할: 방청객 스마트폰에 설치 되어 본 시스템의 Application Server 와 통신하여 재판 멀티미디어 자료를 방청객에게 보여줄 수 있도록 한다.  (QAS-07) 아래와 같은 접근 제어를 통해 보안 기준을 만족 시키도록 하며, MVC 구조 및 Observer 패턴을 적용 하여 (QAS-06) 편리한 사용성 기준을 만족 시킬 수 있는 기반을 마련한다.   * VPN 연결이 되어야만 동작 * 방청객 일련번호 확인시 동작 됨 * 프로그램이 실행 되면 스마트폰 보안 설정이 자동으로 설정됨 * 재판 멀티미디어 자료 조회 가능   Multiplicity: 1..100  특성   * Android Application으로 개발됨 |
| Application Server | 역할: 방청객의 모바일 앱이 사용할 API를 제공 하며, 소송관계자 및 재판부 PC에서 브라우저로 접속할 웹 프로그램도 제공한다. 재판의 시작 부터 종료 시점까지 안정적인 운용을 보장 하기 위해 (QAS-04) 2대의 서버가 Master-Slave 방식으로 동작 한다. 또한 빠른 응답 속도 보장(QAS-02) 을 위해 In Memory Cache 를 가진다.  Multiplicity: 2  특성   * 32 코어 / 32 GM RAM |
| Core Service Server | 역할: 법원TV / 영상 법정 시스템과 같은 외부 시스템과 연동 되면서 디지털 법정 시스템에서 만들어지는 여러 영상을 외부 시스템으로 전달 하는 역할을 한다. 향후 기능 VR 과 같은 기능 추가가 예상되어 (QAS-03) Dispatcher 를 추가하여 확장성에 대비하고, (QAS-01) 영상의 Latency 최소화를 위해 Web RTC를 이용한다. Adapter 패턴을 적용하여 미래에 VR 도입과 같은 기능 확장에 대비한다.  Multiplicity: 1  특성   * 32 코어 / 64 GM RAM |
| Database Management Server | 역할: 시스템이 필요로 하는 재판 정보를 저장하기 위한 데이터베이스 서버이며, 재판의 시작부터 종료 시점까지 안정적인 운용을 보장하기 위해 (QAS-04) 2대의 서버가 Master-Slave 방식으로 동작 한다.  Multiplicity: 2  특성   * 32 코어 / 12GB RAM / 8 TB |
| Court Monitoring Server | 역할: 디지털 법정 시스템의 안정적인 운영과 시스템 장애 탐지 (QAS-05) 를 위해 모니터링 하는 시스템이다.  Multiplicity: 1  특성   * 32 코어 / 64 GM RAM |

### Execution Environment Specification

|  |  |  |
| --- | --- | --- |
| **Node** | **Name** | **Description** |
| Smart Phone | Android 8 | 2023년 5월 데이터 기준 안드로이드 OS 8.0 이상 사용자가 92.4%이다. |
| Judge PC | HTML5 Browser | HTML5 Browser는 다양한 디바이스와 OS에서 동일한 경험을 제공하여 사용자의 사용성을 향상시킬 수 있으며, 데스크탑 Application 설치 번거로움 없이 추후 구현될 추가 기능에 대해서도 간단한 웹 페이지 접근만으로도 확인이 가능하여 확장성이 높다 |
| Litigants PC | HTML5 Browser | HTML5 Browser는 다양한 디바이스와 OS에서 동일한 경험을 제공하여 사용자의 사용성을 향상시킬 수 있으며, 데스크탑 Application 설치 번거로움 없이 추후 구현될 추가 기능에 대해서도 간단한 웹 페이지 접근만으로도 확인이 가능하여 확장성이 높다 |
| Application Server/Core Service Server/Database Management Server/Court Monitoring Server | Ubuntu 20.04 | Ubuntu LTS 지원 버전으로 장기간 지원이 가능  무료 사용 가능  안정적인 버전 |
| Application Server/Core Service Server/Court Monitoring Server | OpenJDK 11 | Tomcat 9를 안정적으로 지원 |
| Application Server/Core Service Server/Court Monitoring Server | Tomcat 9 | 다양한 프로젝트에서 활용되는 WAS  안정적인 운영 가능 |
| Database Management Server | MySQL 8.0 | 관계형 데이터베이스로 널리 사용됨  무료 사용 가능 |

### Communication Path Specification

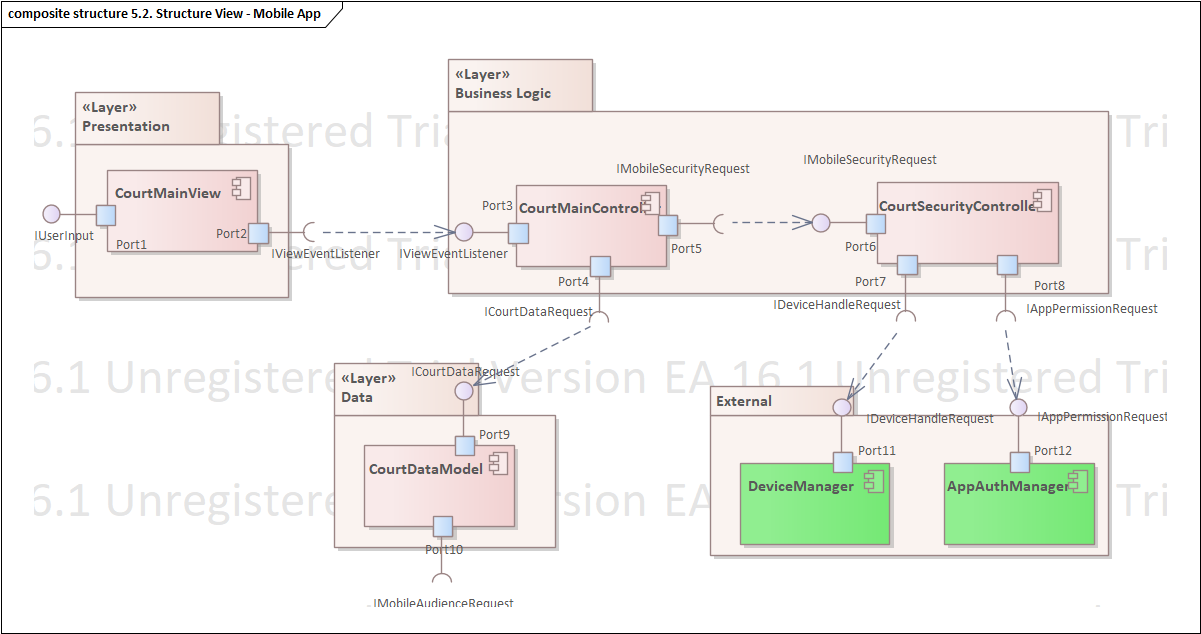
|  |  |
| --- | --- |
| **Path** | **Description** |
| Screens - Application Server | HDMI Cable - 장치간 직접 연결 |
| Multimedia Input Devices - Core Service Server | HDMI Cable - 장치간 직접 연결 |
| Mobile Application - Application Server | HTTPS - 보안 연결 |
| PC in the Court - Application Server | HTTPS - 보안 연결 |
| Court Data Server System - Application Server | HTTPS - 보안 연결 |
| Court Broadcasting System - Core Service Server | HTTPS then RTP/UDP   * HTTPS: Dispatcher 와 통신 * RTP/UDP: 영상 데이터 전송 |
| Online Court System - Core Service Server | HTTPS then RTP/UDP   * HTTPS: Dispatcher 와 통신 * RTP/UDP: 영상 데이터 전송 |
| System Manager SW - Court Monitoring Server | HTTPS - 보안 연결 |
| Application Server - Database Management Server | TCP/IP - 시스템 내부 연결 |
| Application Server - Core Service Server | TCP/IP - 시스템 내부 연결 |
| Application Server - Court Monitoring Server | TCP/IP - 시스템 내부 연결 |
| Database Management Server - Core Service Server | TCP/IP - 시스템 내부 연결 |
| Database Management Server - Court Monitoring Server | TCP/IP - 시스템 내부 연결 |
| Core Service Server - Court Monitoring Server | TCP/IP - 시스템 내부 연결 |

## Structure View

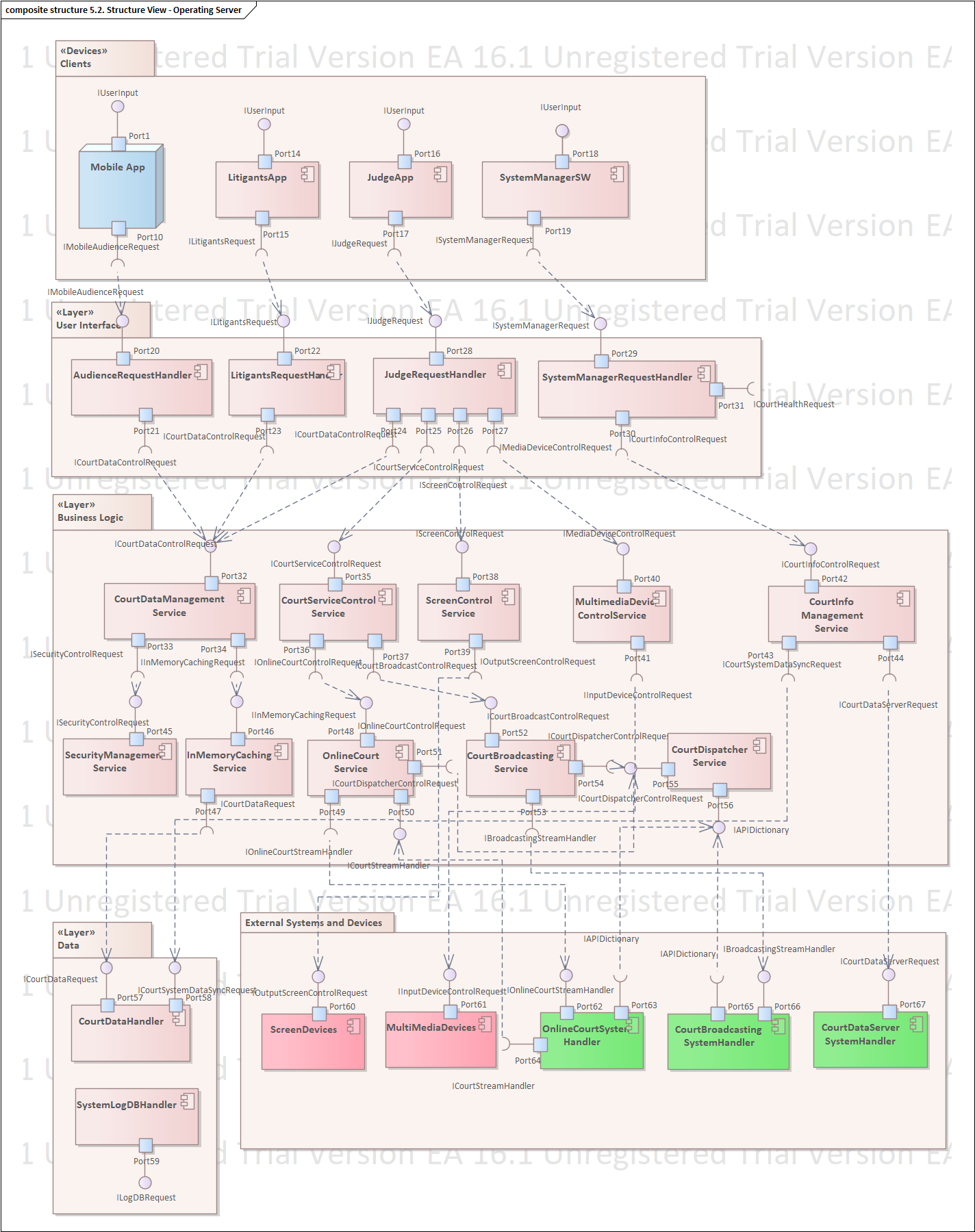
### Static Structure Model

#### Static Structure Diagram

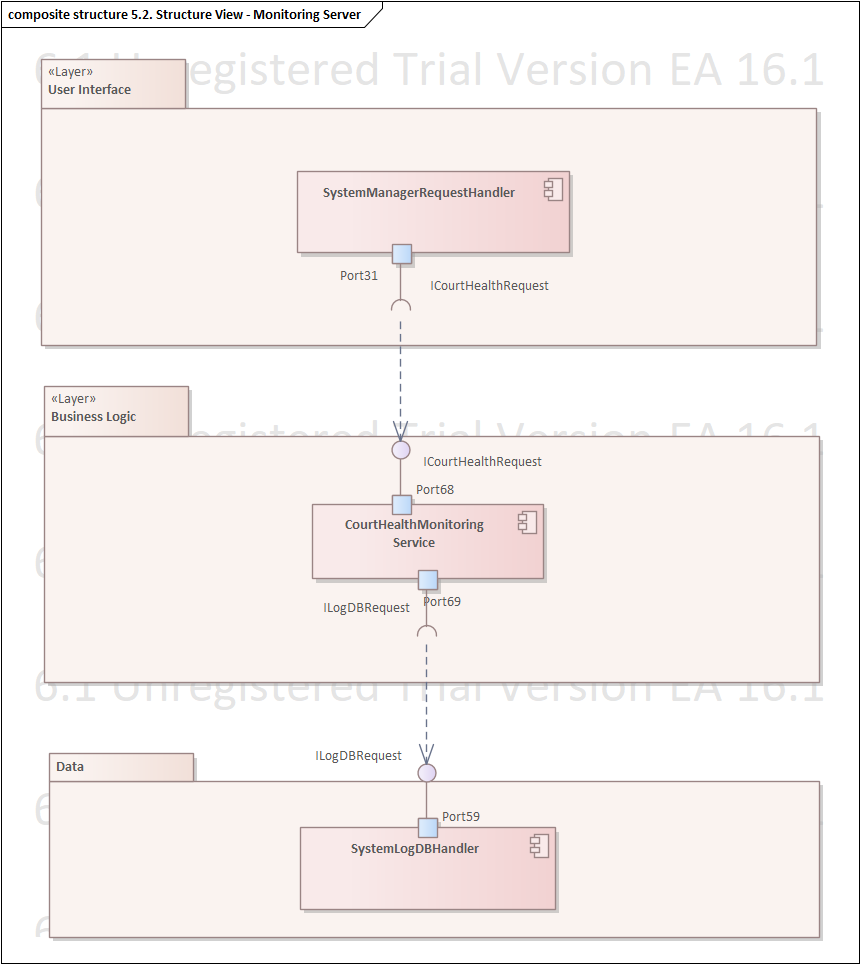
**[Mobile App]**



**[Operating Server]**



**[Monitoring server]**



#### Element List

[Clients]

|  |  |  |
| --- | --- | --- |
| **Name** | **Responsibility** | **Relevant ADs** |
| Mobile App | 방청객 스마트 폰 | - |
| LitigantsApp | 소송관계자 PC | - |
| JudgeApp | 재판부 PC | - |
| SystemManagerSW | 시스템 관리자 SW | - |

[Layers]

|  |  |  |
| --- | --- | --- |
| **Name** | **Responsibility** | **Relevant ADs** |
| Presentation Layer | Mobile App 에서 사용자와 Interaction 하는 UI View Page이다. | - |
| User Interface Layer | 사용자 입력을 내부 비즈니스 로직과 연결 하기 위한 interface layer 이다. | - |
| Business Logic Layer | User Interface Layer로부터 송신한 요청에 대해 알맞은 Service Logic을 수행하는 패키지 | - |
| Data Layer | Service Logic Layer의 요청을 받아 DB에 직접 접근하여 데이터를 처리하는 패키지 | - |

[Mobile App]

|  |  |  |
| --- | --- | --- |
| **Name** | **Responsibility** | **Relevant ADs** |
| CourtMainView | 사용자 입력을 받는 View | UC-01  UC-02  QA-01  QA-04 |
| CourtMainController | View의 비즈니스 로직을 수행하는 모듈 | UC-01  UC-02  QA-01  QA-04 |
| CourtSecurityController | 방청객의 스마트폰 디바이스를 법정 보안 모드로 설정하기 위한 security 모듈 | UC-01  UC-02  QA-05 |
| CourtDataModel | Operating Server와 연동하여 방청객 스마트폰에서 재판 자료가져와 Controller 와 View에게 notify 해 주는 모듈 | UC-01  UC-02  QA-01  QA-04 |

[Operating Server]

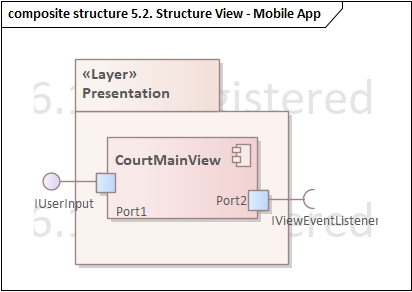
|  |  |  |
| --- | --- | --- |
| **Name** | **Responsibility** | **Relevant ADs** |
| AudienceRequestHandler | 방청객의 입력 요청을 처리 | UC-01  UC-02 |
| LitigantsRequestHandler | 소송관계자의 입력 요청을 처리 | UC-05 |
| JudgeRequestHandler | 재판부의 입력 요청을 처리 | UC-06 |
| SystemManagerRequestHandler | 시스템 관리자 SW의 입력 요청을 처리 | UC-09  UC-10 |
| CourtDataManagementService | 디지털 법정 멀티미디어 자료 관리 | UC-02  UC-05  UC-06 |
| CourtServiceControlService | 디지털 법정 연동 서비스 컨트롤 | UC-03  UC-04 |
| ScreenControlService | 디지털 법정 현장 공용 스크린 컨트롤 | UC-08 |
| MultimediaDeviceControlService | 디지털 법정 현장 멀티미디어 입력 장치 컨트롤 | UC-03  UC-04 |
| SecurityManagementService | 디지털 법정의 접근 제어 보안 컨트롤 | QA-05 |
| InMemoryCachingService | 법정 자료 DB를 캐쉬 하여 관리하며 Preview data 생성 | UC-02  UC-05  UC-06  QA-01  QA-03 |
| OnlineCourtService | 영상 법정 시스템 연동 컨트롤 | UC-04 |
| CourtBroadcastingService | 재판 방송 시스템 연동 컨트롤 | UC-03 |
| CourtDispatcherService | 외부 시스템 인터페이스 컨트롤 | UC-03  UC-04  QA-02 |
| CourtInfoManagementService | 재판 기준 정보 관리 | UC-07  UC-10 |
| CourtDataHandler | 디지털 법정 재판 정보 관리 | UC-10 |
| SystemLogDBHandler | 디지털 법정 시스템 운영 Log 관리 | UC-09 |

[Monitoring Server]

|  |  |  |
| --- | --- | --- |
| **Name** | **Responsibility** | **Relevant ADs** |
| CourtHealthMonitoringService | 디지털 법정 시스템 운영 현황 모니터링 | QA-03 |

### CourtMainView

#### CourtMainView Specification



#### Interface List

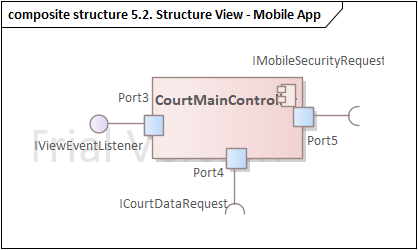
|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *IUserInput* | Provided | 사용자의 Input을 받는 Interface |
| *IViewEventListener* | Required | View의 Event를 Controller로 전달하는 Interface |

#### IUserInput Interface Specification

|  |  |
| --- | --- |
| Operation | Responsibility |
| *IUserInput* | 사용자 UI 입력 |

### CourtMainController

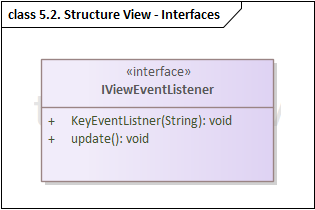
#### CourtMainController Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *IViewEventListener* | Provided | View의 Event를 Controller로 전달하는 Interface |
| *ICourtDataRequest* | Required | 현재 재판 자료와 관련된 data와 관련된 처리를 요청하는 Interface |
| *IMobileSecurityRequest* | Required | 재판 자료를 보기 위하 모바일의 보안 설정을 요청하는 Interface |

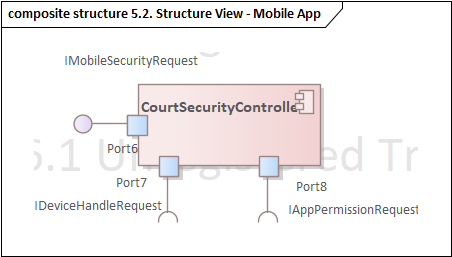
#### *IViewEventListener* Interface Specification



|  |  |
| --- | --- |
| Operation | Responsibility |
| *KeyEventListner()* | View 에서 발생하는 사용자의 Key Event를 Controller 에 전달 하여 필요한 동작을 할 수 있게 한다. |
| *update()* | Model 를 observer로 등록한 이후 데이터 변경에 대한 notify를 받아 변경을 반영하는 함수 |

### CourtSecurityController

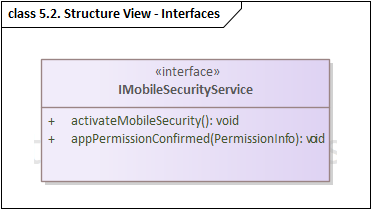
#### CourtSecurityController Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *IMobileSecurityRequest* | Provided | 방청객 스마트폰의 법정 보안 설정을 위해 제공하는 interface |
| *IDeviceHandleRequest* | Required | 스마트폰의 DeviceManager를 통해 기능 조작하는 interface |
| *IAppPermissionRequest* | Required | Mobile App이 Device를 조작할 수 있는 permission을 요청하는 interface |

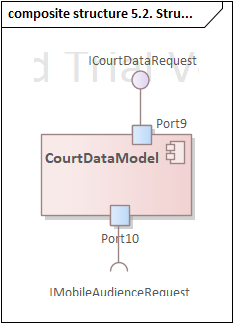
#### IMobileSecurityRequest Interface Specification

****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *appPermissionConfirmed* | 앱이 Device를 조작할 수 있는 permission을 confirm 받는 함수 |
| *activateMobileSecurity* | 앱이 Device의 법정 보안 설정을 activate 시키는 함수 |

### CourtDataModel

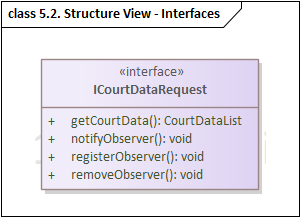
#### CourtDataModel Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *ICourtDataRequest* | Provided | Controller에서 Model로 실제 Court Data 를 요청하는 interface |
| *IMobileAudienceRequest* | Required | Operation Server 에서 방청객 Mobile App이 요청하는 재판 정보를 전달 하기 위한 interface |

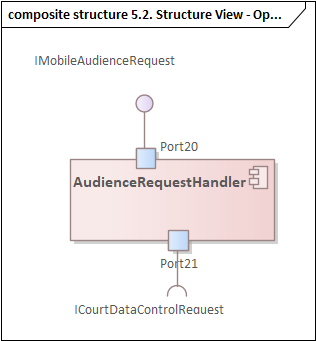
#### ICourtDataRequest Interface Specification

****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *getCourtData()* | 현재 재판 자료와 관련된 data와 관련된 처리를 요청하는 함수 |
| *notifyObserver()* | observer로 등록 된 함수에게 data 변경을 notify 하는 함수 |
| *registerObserver()* | observer 등록 |
| *removeObserver()* | observer 등록 해지 |

### AudienceRequestHandler

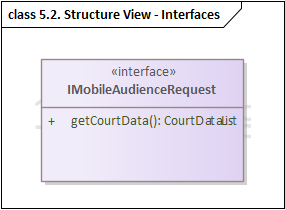
#### AudienceRequestHandler Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *IMobileAudienceRequest* | Provided | 방청객 스마트폰으로 부터 들어오는 입력 interface |
| *ICourtDataControlRequest* | Required | 디지털 재판과 관련된 데이터를 컨트롤 하기 위한 Interface |

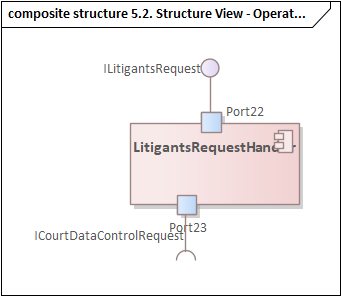
#### IMobileAudienceRequest Interface Specification

****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *getCourtData()* | 방청객 앱에서 재판 멀티미디어 자료 data를 요청하는 함수 |

### LitigantsRequestHandler

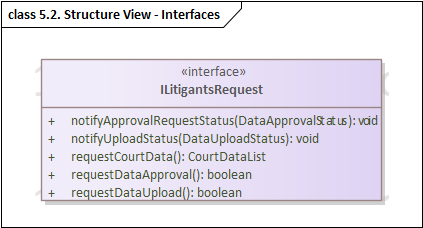
#### LitigantsRequestHandler Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *ILitigantsRequest* | Provided | 소송관계자 PC로 부터 들어올 수 있는 입력 Interface |
| *ICourtDataControlRequest* | Required | 디지털 재판과 관련된 데이터를 컨트롤 하기 위한 Interface |

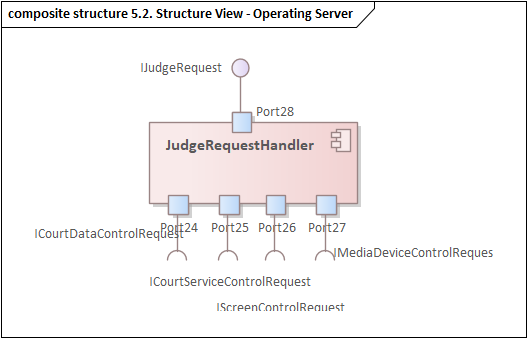
#### ILitigantsRequest Interface Specification



|  |  |
| --- | --- |
| Operation | Responsibility |
| *requestDataUpload()* | PC 에서 시스템에 멀티미디어 자료를 업로드 하는 함수 |
| *requestDataApproval()* | 입력한 파일에 대한 판사의 승인을 요청하는 함수 |
| *requestCourtData()* | 현재 재판과 관련된 멀티미디어 자료를 요청하는 함수 |
| *notifyUploadStatus()* | Upload 현황 확인 함수 |
| *notifyApprovalRequestStatus()* | Approval Request 결과 확인 |

### JudgeRequestHandler

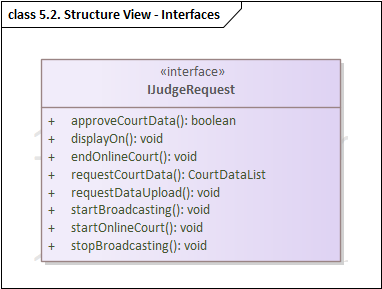
#### JudgeRequestHandler Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *IJudgeRequest* | Provided | 재판부 PC의 Browser를 통해서 재판의 진행/스크린 및 멀티미디어 입력장치 조작, 영상법정/법원TV 조작 등을 진행할 수 있게하는 Interface를 제공한다. |
| *ICourtDataControlRequest* | Required | 재판 자료를 관리하는 Interface |
| *ICourtServiceControlRequest* | Required | 재판의 서비스를 관리 하는 Interface |
| *IScreenControlRequest* | Required | 재판의 공용 스크린을 조작하는 interface |
| *IMediaDeviceControlRequest* | Required | 재판의 멀티미디어 입력 장치를 조작하는 interface |

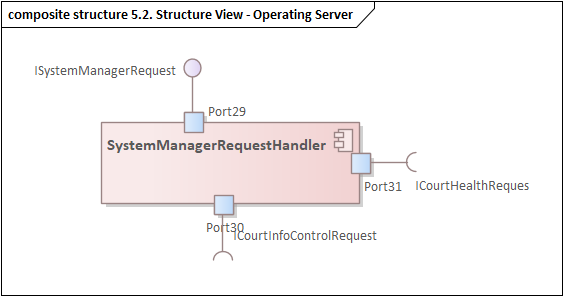
#### IJudgeRequest Interface Specification

****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *stopBroadcasting()* | 재판의 방송을 완료한다. |
| *startOnlineCourt()* | 영상 재판을 시작한다. |
| *startBroadcasting()* | 재판의 방송을 시작한다. |
| *requestDataUpload()* | 현재 재판과 관련된 멀티미디어 자료를 업로드 한다. |
| *requestCourtData()* | 현재 재판과 관련된 멀티미디어 자료 리스트를 요청한다. |
| *endOnlineCourt()* | 영상 재판을 완료한다. |
| *displayOn()* | 멀티미디어 자료중 하나를 선택하여 공용 스크린에 보이도록 한다. |
| *approveCourtData()* | 현재 재판 자료 중 승인 되지 않은 자료를 승인한다. |

### SystemManagerRequestHandler

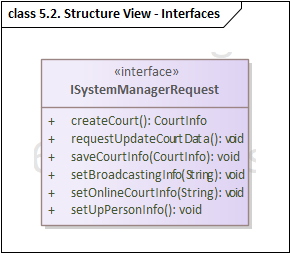
#### SystemManagerRequestHandler Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *ISystemManagerRequest* | Provided | 시스템 매니저가 필요로 하는 작업 들을 진행 할 수 있게 시스템 매니저 SW로 제공하는 Interface |
| *ICourtInfoControlRequest* | Required | 재판의 기준 정보를 설정하기 위한 interface |
| *ICourtHealthRequest* | Required | 현재 진행 중인 디지털 법정 시스템의 상태를 모니터링 하기 위한 interface |

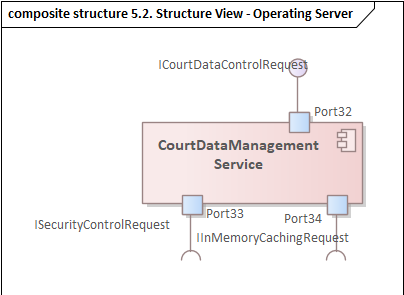
#### ISystemManagerRequest Interface Specification



|  |  |
| --- | --- |
| Operation | Responsibility |
| *requestUpdateCourtData()* | 현재 재판과 관련하여 기존에 존재하는 데이터가 있으면 해당 데이터를 법원 자료 서버로 부터 업데이트 하기 위한 함수 |
| *createCourt()* | 디지털 재판을 생성하는 함수 |
| *setUpPersonInfo()* | 재판과 관련된 인력 정보를 설정하는 함수 |
| *setBroadcastingInfo()* | 재판 방송 진행을 위해 법원TV 시스템으로 부터 받은 방송 ID를 설정하는 함수 |
| *setOnlineCourtInfo()* | 영상 재판 진행을 위해 영상 법정 시스템으로 부터 받은 영상재판 ID를 설정하는 함수 |
| *saveCourtInfo()* | 변경된 디지털 재판 자료를 법원 자료 서버로 저장을 요청하는 함수 |

### CourtDataManagementService

#### CourtDataManagementService Specification

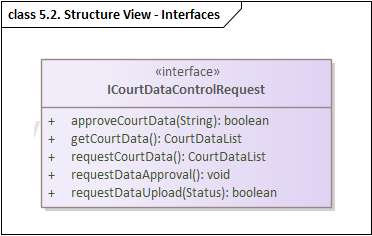


#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *ICourtDataControlRequest* | Provided | 재판 자료를 관리 및 승인을 위한 Interface |
| *ISecurityControlRequest* | Required | 입력의 진입점에 따른 시스템 접근 제어 확인을 위한 Interface |
| *IInMemoryCachingRequest* | Required | 재판 자료 접근 성능 향상을 위해, Database와 In Memory 상에 Cache 및 Preview data 를 관리 할 수 있는 Interface |

#### 

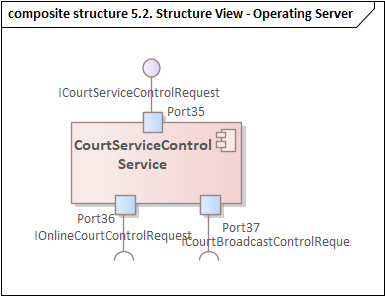
#### ICourtDataControlRequest Interface Specification



|  |  |
| --- | --- |
| Operation | Responsibility |
| *requestDataUpload()* | 재판에 필요한 멀티미디어 자료 업로드 함수 |
| *requestDataApproval()* | 재판 자료 승인 요청 함수 |
| *requestCourtData()* | 업로드된 재판 자료 리스트를 요청하는 함수 |
| *getCourtData()* | 방청객 스마트폰에서 승인된 재판 자료 리스트를 요청하는 함수 |
| *approveCourtData()* | 업로드된 재판 자료 중 하나를 승인하는 함수 |

### CourtServiceControlService

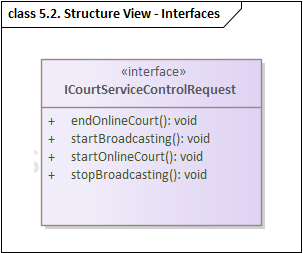
#### CourtServiceControlService Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *ICourtServiceControlRequest* | Provided | 현재 재판과 관련된 서비스를 컨트롤 하기 위한 Interface |
| *IOnlineCourtControlRequest* | Required | 영상재판과 관련된 서비스를 컨트롤 하기 위한 Interface |
| *ICourtBroadcastControlRequest* | Required | 재판 방송 서비스를 컨트롤 하기 위한 interface |

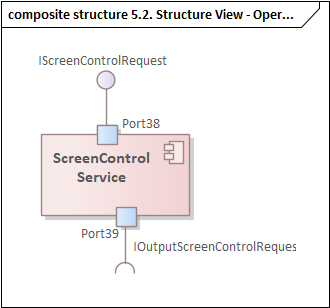
#### ICourtServiceControlRequest Interface Specification



|  |  |
| --- | --- |
| Operation | Responsibility |
| *startBroadcasting()* | 재판의 방송을 시작하는 함수 |
| *stopBroadcasting()* | 재판의 방송을 중지하는 함수 |
| *startOnlineCourt()* | 영상 재판을 시작하는 함수 |
| *endOnlineCourt()* | 영상 재판을 종료하는 함수 |

### ScreenControlService

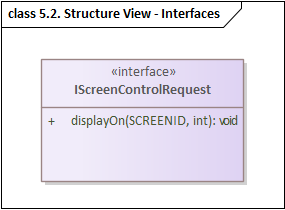
#### ScreenControlService Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *IScreenControlRequest* | Provided | 공용 스크린 조작을 위한 Interface |
| *IOutputScreenControlRequest* | Required | 외부 장치 연동 interface |

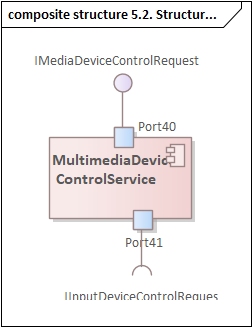
#### IScreenControlRequest Interface Specification



|  |  |
| --- | --- |
| Operation | Responsibility |
| *displayOn()* | SCREENID/ALL 와 Data ID를 입력 받아서 해당 자료를 공용 스크린에 display 하는 함수 |

### MultimediaDeviceControlService

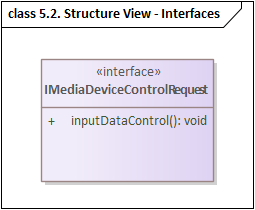
#### MultimediaDeviceControlService Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *IMediaDeviceControlRequest* | Provided | 멀티미디어 입력 Device Control을 위한 Interface |
| *IInputDeviceControlRequest* | Required | 외부 장치에서 제공하는 Device Control Interface |

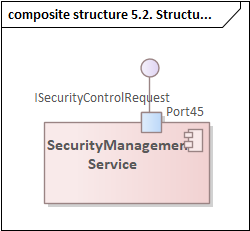
#### IMediaDeviceControlRequest Interface Specification



|  |  |
| --- | --- |
| Operation | Responsibility |
| *inputDataControl()* | 멀티미디어 입력 Device를 Control 하는 함수 |

### SecurityManagementService

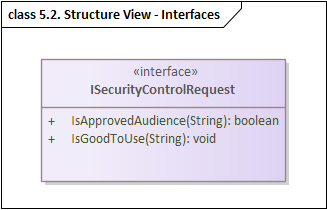
#### SecurityManagementService Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *ISecurityControlRequest* | Provided | 입력의 진입점에 따른 시스템 접근 제어 확인을 위한 Interface |

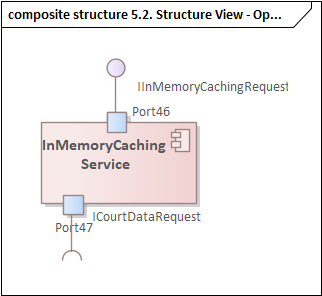
#### ISecurityControlRequest Interface Specification



|  |  |
| --- | --- |
| Operation | Responsibility |
| *IsGoodToUse()* | 해당 함수의 사용이 허용되어 있는지 확인 하는 함수 |
| *IsApprovedAudience()* | 방청객의 일련번호가 현재 재판에 승인되어 있는지 확인 하는 함수 |

### InMemoryCachingService

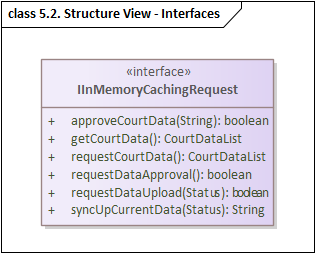
#### InMemoryCachingService Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *IInMemoryCachingRequest* | Provided | 디지털 재판의 자료가 변경이 발생 하였을 때, DB와 Sync 및 service application 내 caching 과 preview data 생성을 위한 interface |
| *ICourtDataRequest* | Required | Database의 조작을 위한 interface |

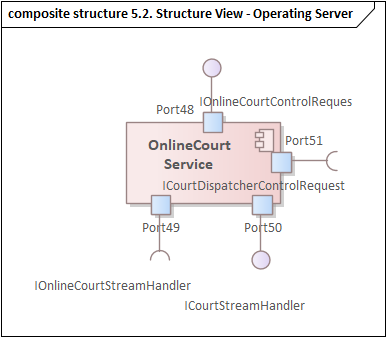
#### IInMemoryCachingRequest Interface Specification

****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *syncUpCurrentData()* | DB to In Memory Sync 함수 |
| *requestDataUpload()* | PC 에서 시스템에 멀티미디어 자료를 업로드 하는 함수 |
| *requestDataApproval()* | 입력한 파일에 대한 판사의 승인을 요청하는 함수 |
| *requestCourtData()* | 현재 재판과 관련된 멀티미디어 자료를 요청하는 함수 |
| *getCourtData()* | 방청객 스마트폰에서 승인된 재판 자료 리스트를 요청하는 함수 |
| *approveCourtData()* | 업로드된 재판 자료 중 하나를 승인하는 함수 |

### OnlineCourtService

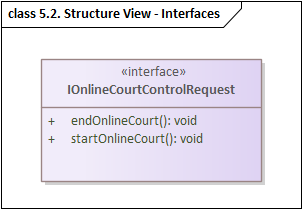
#### OnlineCourtService Specification



#### Interface List

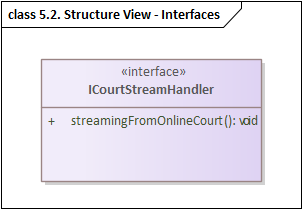
|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *IOnlineCourtControlRequest* | Provided | 영상법정 진행 관리 Interface |
| *ICourtStreamHandler* | Provided | 영상 법정 영상 데이터를 영상 법정 시스템으로 받는 Interface |
| *ICourtDispatcherControlRequest* | Required | 본시스템과 외부 서비스 시스템의 interface 등록을 위한 Interface |
| *IOnlineCourtStreamHandler* | Required | 영상 법정 시스템에서 제공하는 디지털 법정 영상 전송을 위한 외부 Interface |

#### IOnlineCourtControlRequest Interface Specification

****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *startOnlineCourt()* | 영상 법정 재판을 시작 하는 함수 |
| *endOnlineCourt()* | 영상 법정 재판을 종료 하는 함수 |

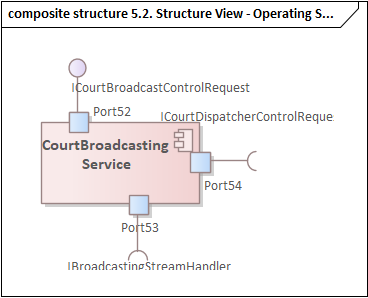
#### ICourtStreamHandler Interface Specification

****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *streamingFromOnlineCourt* | 영상 법정 시스템으로 부터 영상 스트림을 받는 함수 |

### CourtBroadcastingService

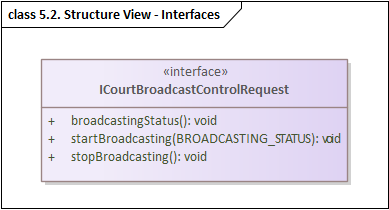
#### CourtBroadcastingService Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *ICourtBroadcastControlRequest* | Provided | 재판의 방송을 Control 하는 interface |
| *ICourtDispatcherControlRequest* | Required | 재판 방송 서비스의 인터페이스를 외부 법원TV 시스템이 확인 할 수 있도록 Dispatcher에 등록하는 interface |
| *IBroadcastingStreamHandler* | Required | 외부 법원TV 시스템이 제공하는 interface로 재판의 영상을 방송을 위해서 전달하는 interface |

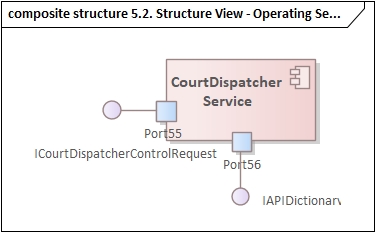
#### ICourtBroadcastControlRequest Interface Specification

****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *startBroadcasting()* | 재판 방송을 시작하는 함수 |
| *stopBroadcasting()* | 재판 방송을 중단하는 함수 |
| *broadcastingStatus()* | 재판 방송의 상태 정보를 제공 하는 함수 |

### CourtDispatcherService

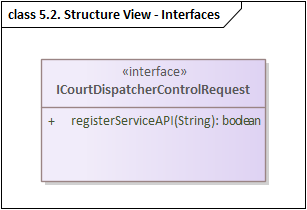
#### CourtDispatcherService Specification



#### Interface List

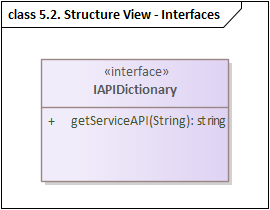
|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *ICourtDispatcherControlRequest* | Provided | 본시스템과 외부 서비스 시스템의 interface 등록을 위한 Interface |
| *IAPIDictionary* | Provided | 외부 서비스들이 본 시스템과 연동하기 위해 어떤 서버와 API를 써야하는지 확인 할 수 있는 Interface |

#### ICourtDispatcherControlRequest Interface Specification

****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *registerServiceAPI()* | 외부로 연동될 API list를 등록하는 함수 |

#### IAPIDictionary Interface Specification

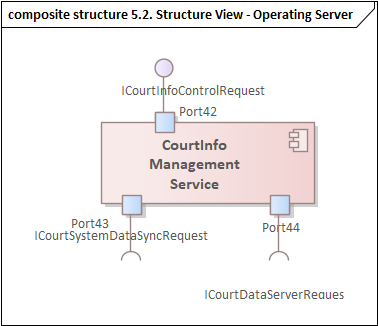


|  |  |
| --- | --- |
| Operation | Responsibility |
| *getServiceAPI()* | 외부 시스템이 본 시스템 내부의 서버와 연동하기 위해 필요한 API를 query 해 갈 수 있는 함수 |

### 

### CourtInfoManagementService

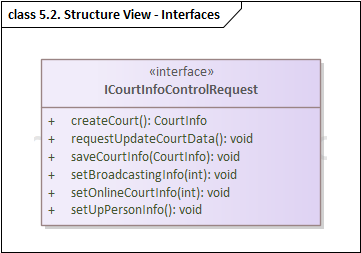
#### CourtInfoManagementService Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *ICourtInfoControlRequest* | Provided | 재판의 기준정보를 세팅하기 위한 Interface |
| *ICourtSystemDataSyncRequest* | Required | 외부 법원 자료 서버 시스템에서 제공하는 Interface |
| *ICourtDataServerRequest* | Required | 본 시스템의 Database에 Data update를 하기 위한 interface |

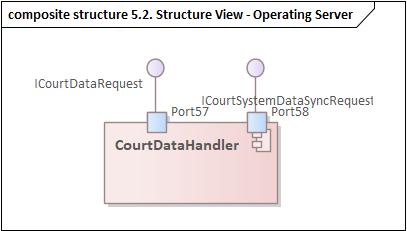
#### ICourtInfoControlRequest Interface Specification

****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *createCourt()* | 디지털 법정을 생성하고 해당 ID를 리턴 하는 기능 |
| *requestUpdateCourtData()* | 현재 재판과 관련하여 기존에 존재하는 데이터가 있으면 해당 데이터를 법원 자료 서버로 부터 업데이트 하기 위한 함수 |
| *saveCourtInfo()* | 변경된 디지털 재판 자료를 법원 자료 서버로 저장을 요청하는 함수 |
| *setBroadcastingInfo()* | 재판 방송 진행을 위해 법원TV 시스템으로 부터 받은 방송 ID를 설정하는 함수 |
| *setOnlineCourtInfo()* | 영상 재판 진행을 위해 영상 법정 시스템으로 부터 받은 영상재판 ID를 설정하는 함수 |
| *setUpPersonInfo()* | 재판과 관련된 인력 정보를 설정하는 함수 |

### CourtDataHandler

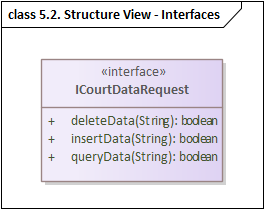
#### CourtDataHandler Specification



#### Interface List

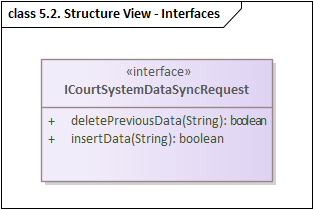
|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *ICourtDataRequest* | Provided | 디지털 법정 진행과 관련된 정보를 DB에 업데이트 하는 Interface |
| *ICourtSystemDataSyncRequest* | Provided | 법원 자료 서버 시스템과 재판의 기준 정보를 Sync 하기 위한 Interface |

#### ICourtDataRequest Interface Specification

****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *queryData()* | DB 에서 특정 data를 query 하는 함수 |
| *insertData()* | DB 에 특정 data를 insert 하는 함수 |
| *deleteData()* | DB 에 있는 data를 삭제하는 함수 |

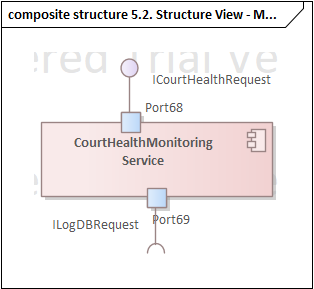
#### ICourtSystemDataSyncRequest Interface Specification

****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *insertData()* | 법원자료서버 시스템에서 재판의 기존 정보를 가져와 본 시스템의 DB 에 data를 insert 하는 함수 |
| *deletePreviousData()* | 법원자료서버 시스템에서 재판의 기존 정보를 가져와 본 시스템의 DB에 입력하기 위해 기존에 있는 동일 data를 삭제하는 함수 |

### CourtHealthMonitoringService

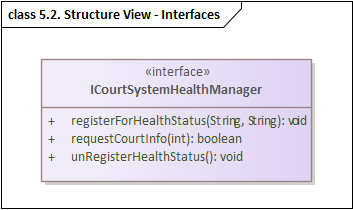
#### CourtHealthMonitoringService Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *ICourtHealthRequest* | Provided | 본 시스템의 운영 상태를 점검하기 위한 interface |
| *ILogDBRequest* | Required | 본 시스템의 운영 상태 log를 관리 하기 위한 interface |

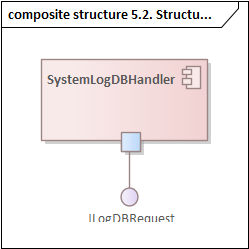
#### ICourtHealthRequest Interface Specification

****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *requestCourtInfo()* | 모니터링할 재판 정보 확인 요청 함수 |
| *registerForHealthStatus()* | 재판 시스템 동작 Health 상태 확인을 위한 상태 등록 함수 |
| *unRegisterHealthStatus()* | 재판 시스템 동작 Health 상태 등록을 받지 않기 위한 해지 함수 |

### SystemLogDBHandler

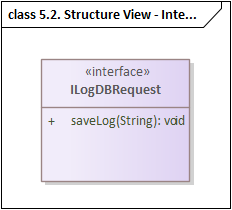
#### SystemLogDBHandler Specification



#### Interface List

|  |  |  |
| --- | --- | --- |
| **Name** | **Kind** | **Description** |
| *ILogDBRequest* | Provided | 본 시스템의 운영 상태 log를 관리 하기 위한 interface |

#### ILogDBRequest Interface Specification

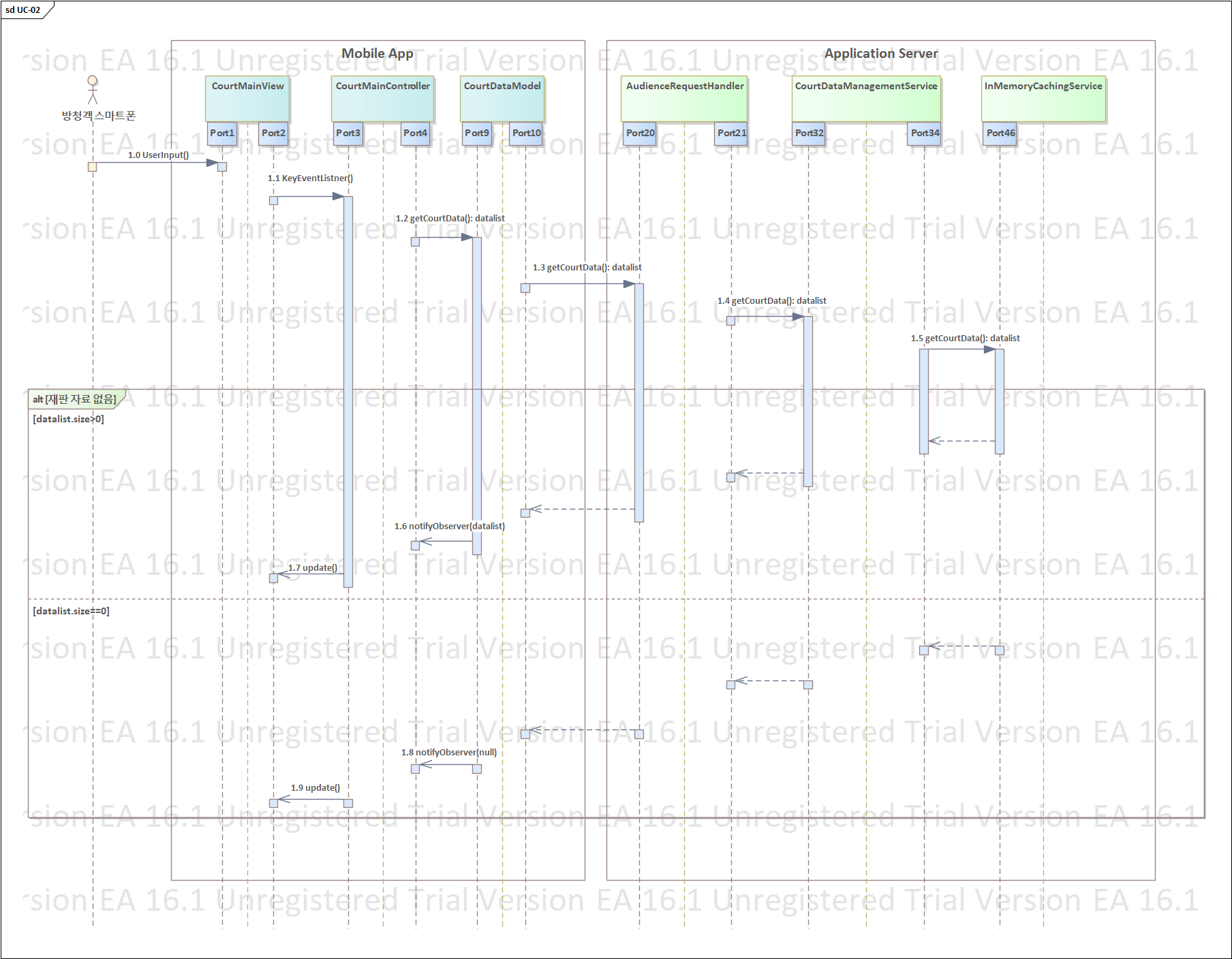
****

|  |  |
| --- | --- |
| Operation | Responsibility |
| *saveLog()* | 본 시스템의 운영 상태 log 저장 |

## Behavior View

### *UC-02 스마트폰 재판 자료 조회*

#### Behavior Diagram



#### Behavior Description

위 Behavior Diagram은 방청객이 스마트폰을 통해서 재판 자료를 조회 하는 과정의 Use Case 시나리오이다.

(1.0) 방청객은 스마트폰의 디지털 법정 앱의 UI를 통해 User Input을 진행 한다. 이를 통해 CourtMainView 와 Interaction을 시작한다.

(1.1) CourtMainController 는 KeyEventListner를 통해 사용자 입력을 확인 하고, 재판 자료 조회 함수를 수행 한다.

(1.2) CourtMainController는 getCourtData() 함수를 통해 CourtDataModel로 자료 조회를 시도한다.

(1.3)CourtDataModel는 HTTPS 로 Operating Server에 접근하여 방청객의 request를 처리하는 AudienceRequestHandler의 getCourtData() 함수를 호출한다.

(1.4) AudienceRequestHandler는 CourtDataManagementService 에서 제공하는 getCourtData()를 호출한다.

(1.5) CourtDataManagementService 는 InMemoryCachingService에서 제공하는 getCourtData를 호출하고, Database 까지 query 하지 않고 caching된 data가 있음을 확인 후 return 받는다. 즉, Maintain multiple copy of data를 통해 **[QA-01] Performance -** 모바일 앱 사용자의 응답속도를 보장한다.

(1.6) CourtDataModel는 가져온 datalist가 변경 되었음을 확인하고 CourtMainController 에게 notifyObserver 함수로 알려준다. 즉 MVC 구조를 적용하여 향후 앱의 사용성을 위한 개발이 더욱 유연하게 진행 될 수있게 한다. **[QA-04] Usability**

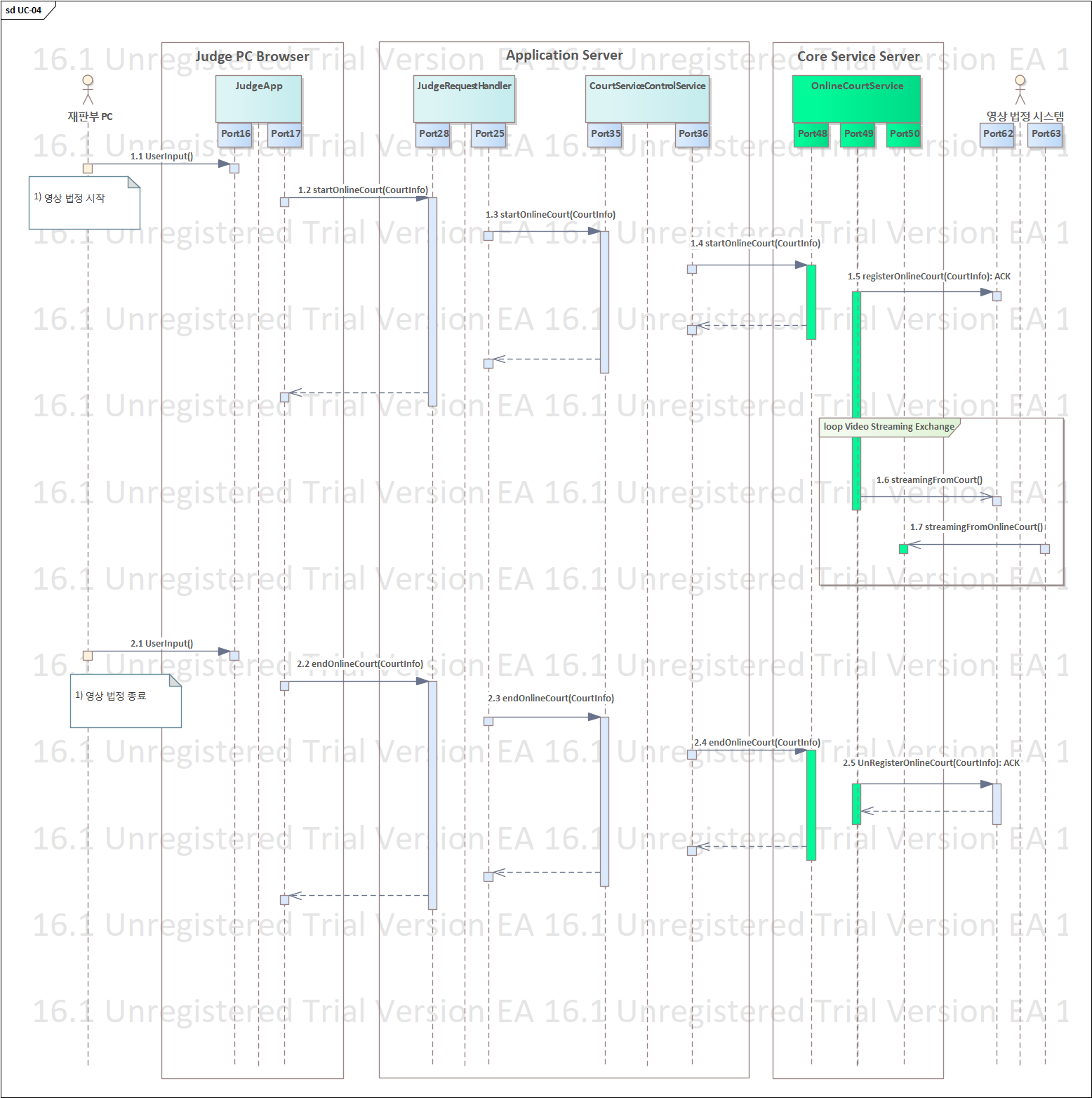
(1.7) CourtMainController 는 가져온 datalist가 변경 되었음을 확인하고 CourtMainView 에게 update함수로 알려준다.

(1.8) CourtDataModel는 가져온 datalist가 null로 변경 되었음을 확인하고 CourtMainController 에게 notifyObserver 함수로 알려준다.

(1.9) CourtMainController 는 가져온 datalist가 null로 변경 되었음을 확인하고 CourtMainView 에게 update 함수로 알려준다.

### *UC-04 영상 법정 연동*

#### Behavior Diagram



#### Behavior Description

위 Behavior Diagram은 재판부가 PC를 통해 영상 재판을 시작하고 완료 하는 과정의 Use Case 시나리오이다.

(1.1) 재판부 PC 에서 browser를 통해 UserInput을 통해 시스템에 온라인 **영상 재판 시작을 요청**한다.

(1.2) JudgeRequestHandler는 startOnlineCourt를 통해 영상 재판 요청을 받고 서비스 컨트롤을 위해 CourtServiceControlService에 넘긴다.

(1.3) CourtServiceControlService는 startOnlineCourt를 통해 영상 재판 요청을 받고 영상 재판 컴포넌트인 OnlineCourtService에 넘긴다.

(1.4) OnlineCourtService는 startOnlineCourt를 통해 영상재판 시작 요청을 받고, 외부 영상재판 시스템과 연동을 시작한다.

(1.5) OnlineCourtService는 registerOnlineCourt 로 영상 스트리밍을 받기 위한 api 세팅을 한다.

(1.6) OnlineCourtService는 외부 영상 재판 시스템에서 제공하는 streamingFromCourt 에 디지털 법정에서 생성된 재판 영상을 전송한다. 1.6은 영상 재판이 완료될 때까지 반복한다.

(1.7) OnlineCourtService는 streamingFromOnlineCourt 를 통해서 외부 영상 재판 시스템으로 부터 들어오는 영상을 시스템에 가져온다. 1.7은 영상 재판이 완료될 때까지 반복한다.

(2.1) 재판부 PC 에서 browser를 통해 UserInput을 통해 시스템에 온라인 **영상 재판 종료를 요청**한다.

(2.2) JudgeRequestHandler는 endOnlineCourt를 통해 영상 재판 요청을 받고 서비스 컨트롤을 위해 CourtServiceControlService에 넘긴다.

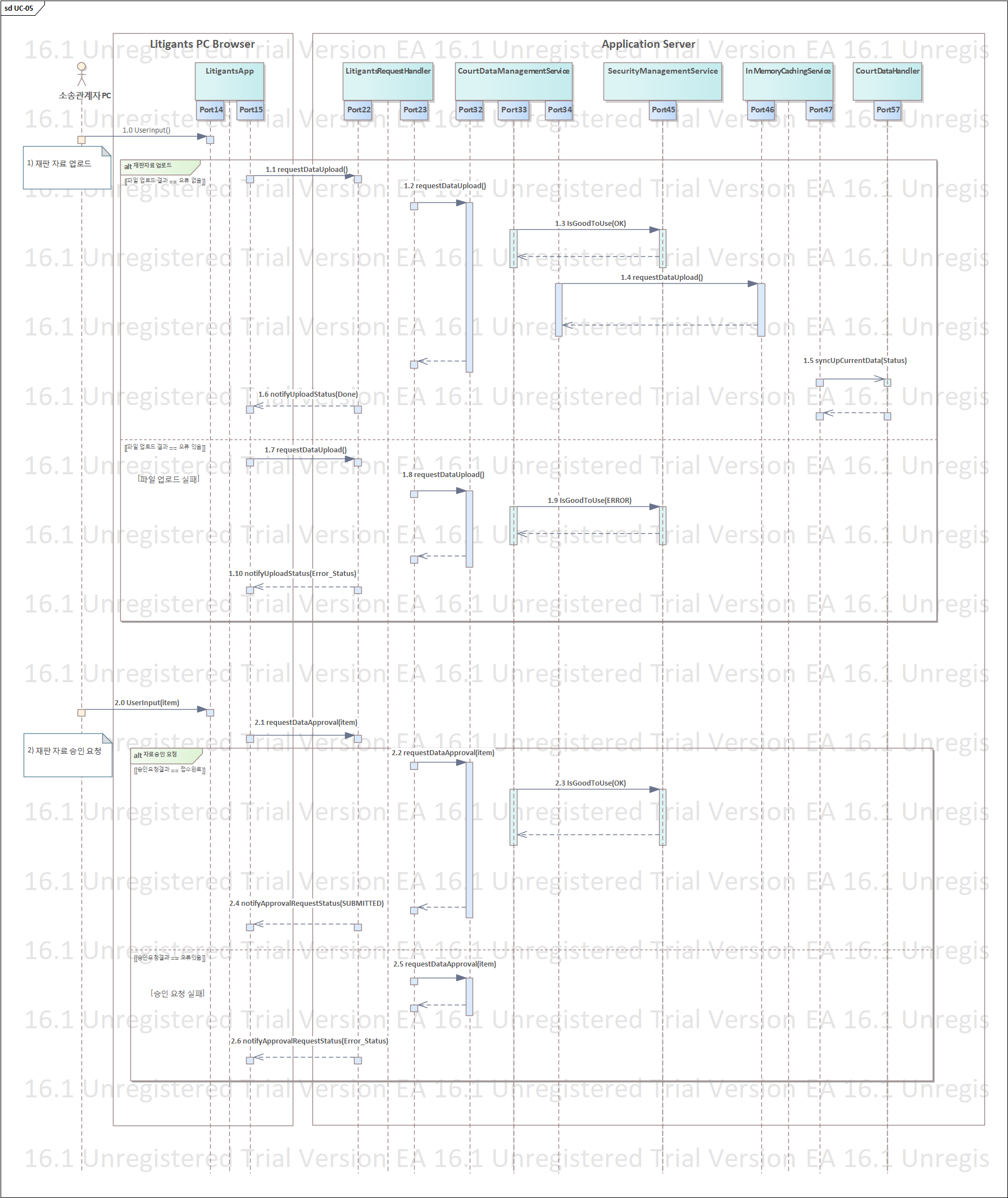
(2.3) CourtServiceControlService는 endOnlineCourt를 통해 영상 재판 요청을 받고 영상 재판 컴포넌트인 OnlineCourtService에 넘긴다.

(2.4) OnlineCourtService는 endOnlineCourt를 통해 영상재판 종료 요청을 받는다.

(2.5) OnlineCourtService는 외부 영상재판 시스템에서 제공하는 api인 UnRegisterOnlineCourt 를 통해 영상 재판의 완료를 통보하고 영상재판을 종료한다.

### *UC-05 자료 승인 요청*

#### Behavior Diagram



#### Behavior Description

위 Behavior Diagram은 소송 관계인이 PC를 통해 자료를 업로드 하고 승인 요청하는 과정의 Use Case 시나리오이다.

(1.0) 소송관계자가 PC를 통해 LitigantsApp 에 data 업로드를 요청한다.

(1.1) LitigantsRequestHandler는 requestDataUpload를 통해 LitigantsApp 로부터 파일 업로드 요청을 받는다.

(1.2) CourtDataManagementService 는 requestDataUpload를 통해 LitigantsRequestHandler로 부터 파일 업로드 요청을 받는다.

(1.3) **SecurityManagementService**는 IsGoodToUse 를 통해 CourtDataManagementService 로부터 **접근 제어 확인**을 요청 받고 이에 대한 결과를 return (OK) 한다. **[QA-05] Security**

(1.4) InMemoryCachingService requestDataUpload를 통해 CourtDataManagementService 로 부터 자료 업로드 요청을 받아 진행한다.

(1.5) CourtDataHandler는 syncUpCurrentData를 통해 InMemoryCachingService 로 부터 업로드 받은 자료를 받아 Database와 sync 한다. **이때 syncUpCurrentData 는 async 하게 진행되어 이전 흐름을 방해 하지 않는다.**

(1.6) LitigantsRequestHandler는 업로드 결과(Done)를 notifyUploadStatus 함수를 통해 전달 받는다.

(1.7) LitigantsRequestHandler는 requestDataUpload를 통해 LitigantsApp 로부터 파일 업로드 요청을 받는다.

(1.8) CourtDataManagementService 는 requestDataUpload를 통해 LitigantsRequestHandler로 부터 파일 업로드 요청을 받는다.

(1.9) SecurityManagementService는 IsGoodToUse 를 통해 CourtDataManagementService 로부터 **접근 제어 확인을 요청** 받고 이에 대한 결과를 return (ERROR) 한다. **[QA-05] Security**

(1.10) LitigantsRequestHandler는 업로드 결과(**Error\_Status**)를 **notifyUploadStatus** 함수를 통해 전달 받는다.

(2.0) 소송관계자가 PC를 통해 LitigantsApp 에게 **자료 승인을 요청한다**.

(2.1) LitigantsRequestHandler는 **requestDataApproval**를 통해 LitigantsApp 로부터 자료 승인 요청을 받는다.

(2.2) CourtDataManagementService 는 **requestDataApproval**를 통해 LitigantsRequestHandler로 부터 자료 승인 요청을 받는다.

(2.3) SecurityManagementService는 IsGoodToUse 를 통해 CourtDataManagementService 로부터 **접근 제어 확인을 요청** 받고 이에 대한 결과를 return (OK) 한다. **[QA-05] Security**

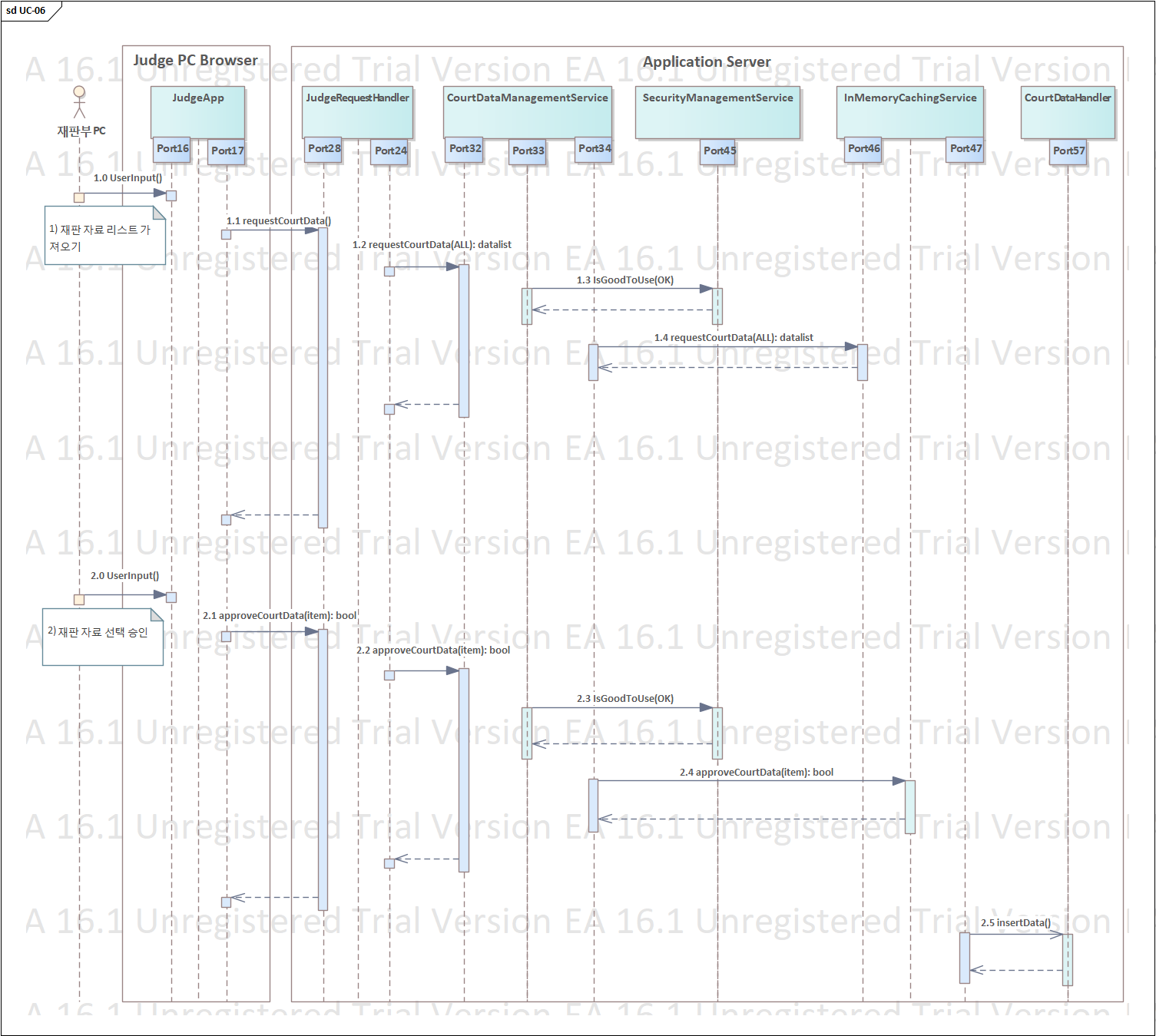
(2.4) LitigantsRequestHandler는 **notifyApprovalRequestStatus 함수**를 통해 CourtDataManagementService 로 부터승인 요청이 submit 되었음을 통보 받는다.

(2.5) CourtDataManagementService 는 **requestDataApproval**를 통해 LitigantsRequestHandler로 부터 자료 승인 요청을 받는다.

(2.6) LitigantsRequestHandler는 **notifyApprovalRequestStatus** 함수를 통해 CourtDataManagementService 로 부터승인 요청이 **SUBMIT\_ERROR**되었음을 통보 받는다.

### *UC-06 재판부 자료 승인*

#### Behavior Diagram



#### Behavior Description

위 Behavior Diagram은 재판부가 PC를 통해 업로드된 재판 자료를 조회 하고, 그 중 하나를 승인 하는 과정의 Use Case 시나리오이다.

(1.1) 재판부는 PC를 통해 재판 자료 조회를 요청 한다.

(1.2) CourtDataManagementService 는 requestCourtData를 통해 JudgeRequestHandler로부터 재판 자료 조회 요청을 받는다.

(1.3) SecurityManagementService 는 IsGoodToUse 를 통해 CourtDataManagementService 로 부터 **접근 제어 확인을 요청** 받고 결과를 리턴(OK) 한다. **[QA-05] Security**

(1.4) InMemoryCachingService는 requestCourtData(ALL) 을 통해 CourtDataManagementService 로 부터 현재 재판과 관련된 모든 자료를 요청 받고 datalist를 리턴 한다.

(2.1) 재판부는 PC를 통해 재판 자료 승인을 요청 한다.

(2.2) CourtDataManagementService 는 approveCourtData(item)을 통해 JudgeRequestHandler로부터 재판 자료 승인 요청을 받는다.

(2.3) SecurityManagementService 는 IsGoodToUse 를 통해 CourtDataManagementService 로 부터 접근 제어 확인을 요청 받고 결과를 리턴(OK) 한다.

(2.4) InMemoryCachingService는 approveCourtData(item) 을 통해 CourtDataManagementService 로 부터 재판 자료 승인 상태를 요청 받고 이를 저장한다.

(2.5) CourtDataHandler는 InMemoryCachingService로 부터 변경된 재판 자료 상태를 async 하게 입력받고 이를 저장 한다.

### *UC-07 법원 자료 업데이트*

#### Behavior Diagram



#### Behavior Description

위 Behavior Diagram은 시스템관리자가 시스템 관리 SW를 통해 법원 자료 서버 시스템에 있는 기존 재판 정보를 디지털 법정 서버로 가져와 기준 정보를 업데이트 시키는 과정의 Use Case 시나리오이다.

(1.1) 시스템관리 SW는 SystemManagerRequestHandler 를 통해 디지털 법정 재판의 기준 정보 업데이트를 요청한다.

(1.2) CourtInfoManagementService 는 SystemManagerRequestHandler로 부터 requestUpdateCourtData 를 요청 받는다.

(1.3) CourtInfoManagementService 는 외부 법원 자료 서버 시스템에서 제공하는 getCourtData 함수를 호출하여 courtData를 받는다.

(1.4) CourtInfoManagementService 는 CourtDataHandler 에 저장되어 있던 기존 자료를 deletePreviousData를 통해 삭제 한다.

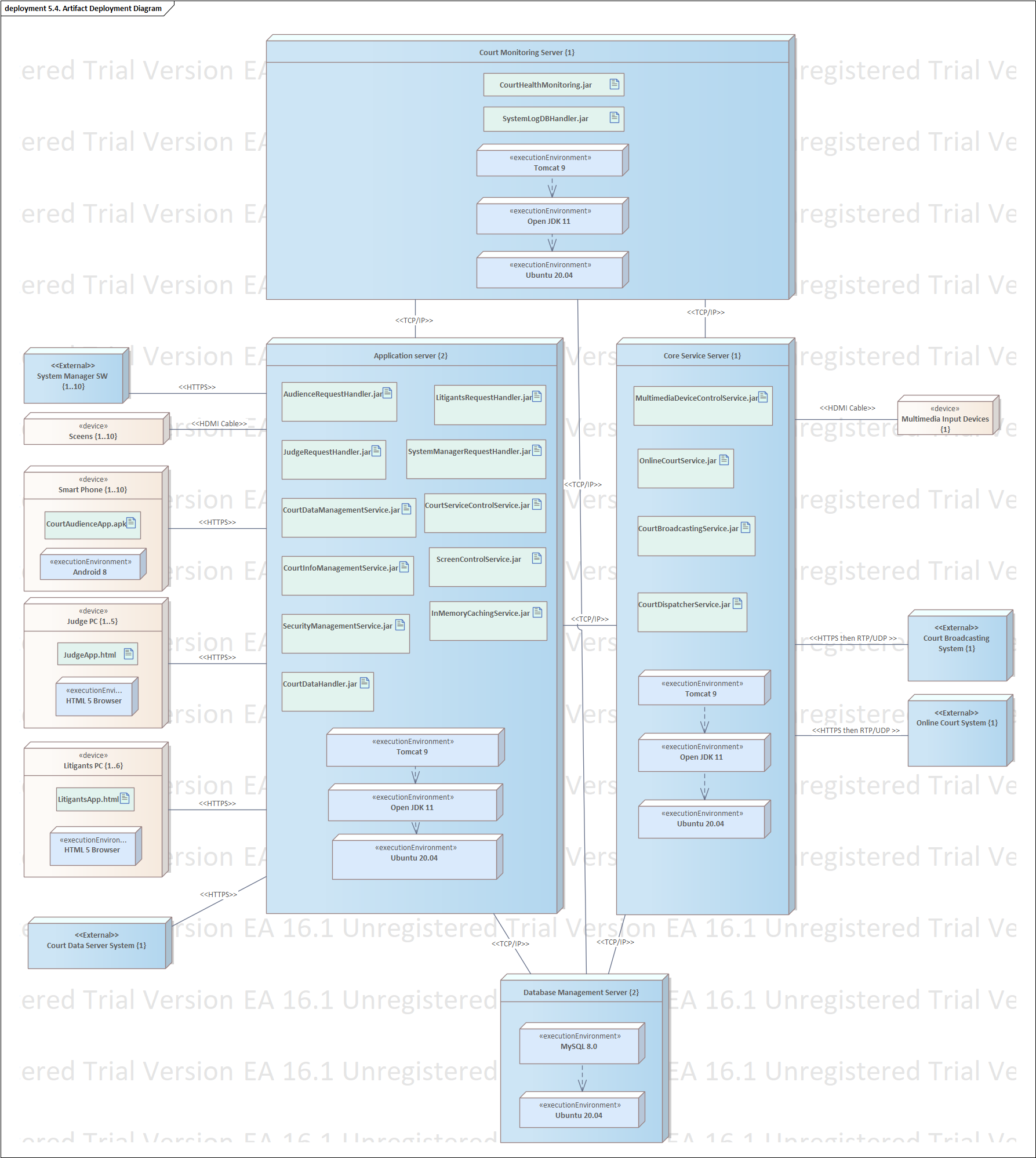
(1.5) CourtInfoManagementService 는 insertData를 사용해 통해 새로 받은 courtInfo를 CourtDataHandler 에 전달 하고 저장한다.

(1.6) SystemManagerRequestHandler는 notifyUpdate 함수를 통해 업데이트가 완료 되었음을 시스템관리 SW에 통보한다.

## Deployment View

### Artifact Deployment Model

#### Artifact Deployment Diagram

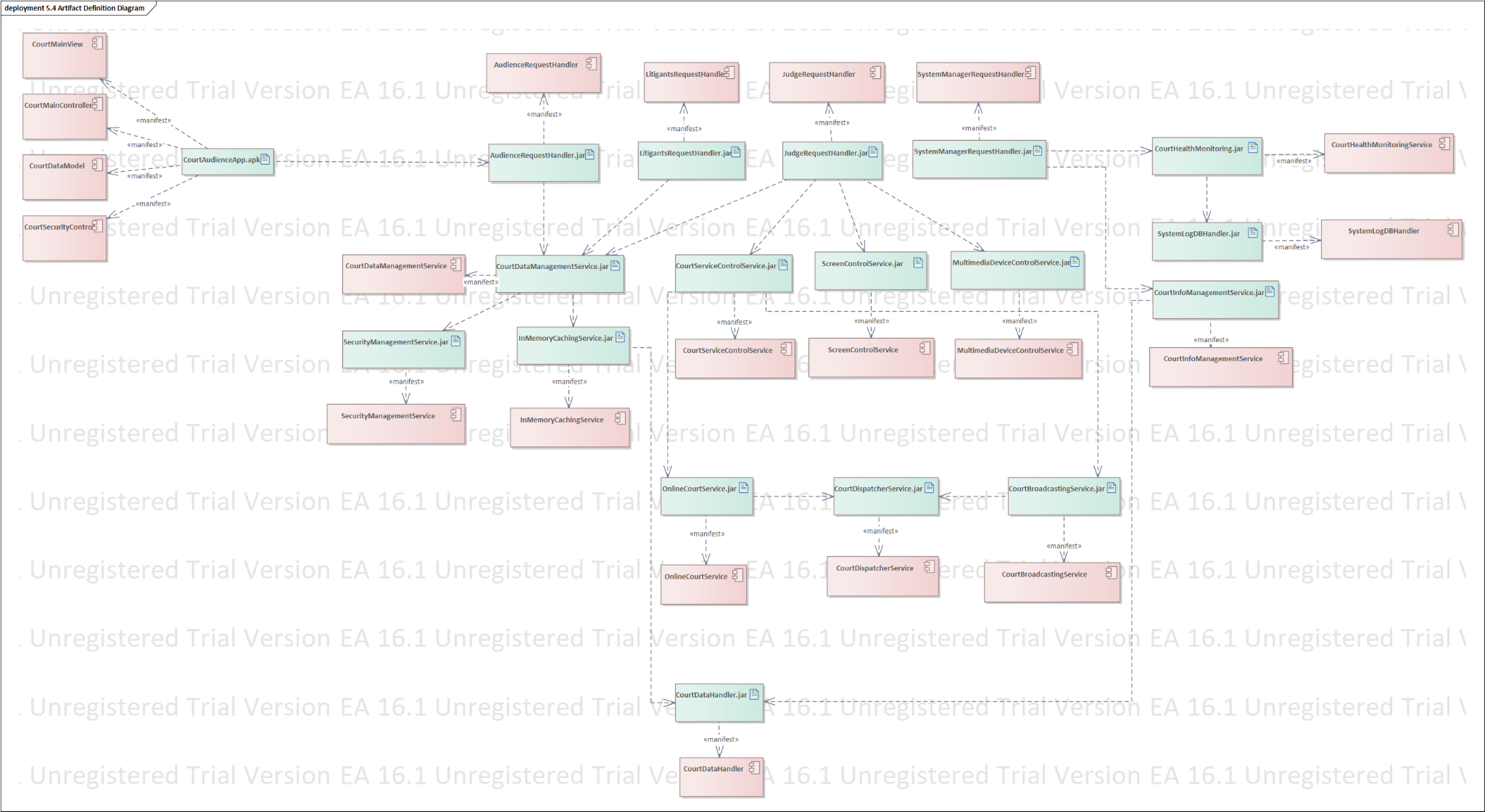
****

#### Artifact Deployment Specification

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Deployment Target** | | **Description and Rationale** |
| **Node** | **Execution Environment** |
| *CourtAudienceApp.apk* | 방청객 스마트폰 | Android 8 | apk 파일을 스마트폰에 설치 |
| *JudgeApp.html* | 재판부 PC | HTML5 Browser | html 파일을 HTML5 Web Browser |
| *LitigantsApp.html* | 소송관계인 PC | HTML5 Browser | html 파일을 HTML5 Web Browser |
| *AudienceRequestHandler* | Application Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *LitigantsRequestHandler* | Application Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *JudgeRequestHandler* | Application Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *SystemManagerRequestHandler* | Application Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *CourtDataManagementService* | Application Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *CourtServiceControlService* | Application Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *ScreenControlService* | Application Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *SecurityManagementService* | Application Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *InMemoryCachingService* | Application Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *CourtInfoManagementService* | Application Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *CourtDataHandler* | Application Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *MultimediaDeviceControlService* | Core Service Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *OnlineCourtService* | Core Service Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *CourtBroadcastingService* | Core Service Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *CourtDispatcherService* | Core Service Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *CourtHealthMonitoringService* | Court Monitoring Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |
| *SystemLogDBHandler* | Court Monitoring Server | Tomcat 9 | jar 파일을 Tomcat 9에 배포 |

### Artifact Definition Model

#### Artifact Definition Diagram



#### Artifact Definition Specification

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Manifested Component** | **Depends on** | **Description and Rationale** |
| *CourtAudienceApp.apk* | *CourtMainView*  *CourtMainController*  *CourtDataModel*  *CourtSecurityController* | AudienceRequestHandler.jar | 디지털 법정의 VPN 망에 들어온, 방청객의 스마트폰에서 동작되는 Mobile App으로 사용자의 입력을 받아 재판 정보를 조회 할 수 있는 기능을 제공한다.  재판 관련 자료 조회를 위해서는 방청객 일련 번호를 입력 해야 하며, 법정 보안 모드 설정을 완료 하고 나면 확인 할 수 있다.  MVC 패턴과 Observer 패턴을 적용하여 사용성 UI 개발을 유연하게 대응할 수 있는 기반 마련 |
| *AudienceRequestHandler.jar* | *AudienceRequestHandler* | CourtDataManagementService.jar | 방청객의 입력 요청을 처리 |
| *JudgeRequestHandler.jar* | *JudgeRequestHandler* | CourtDataManagementService.jar  CourtServiceControlService.jar  ScreenControlService.jar | 재판부의 입력 요청을 처리 |
| *CourtDataManagementService.jar* | *CourtDataManagementService* | SecurityManagementService.jar  InMemoryCachingService.jar | 디지털 법정 멀티미디어 자료 관리 |
| *CourtInfoManagementService.jar* | *CourtInfoManagementService* | CourtDataHandler.jar | 재판 기준 정보 관리 |
| *SecurityManagementService.jar* | *SecurityManagementService* | - | 디지털 법정의 접근 제어 보안 컨트롤 |
| *CourtDataHandler.jar* | *CourtDataHandler* | - | 디지털 법정 재판 정보 관리 |
| *LitigantsRequestHandler.jar* | *LitigantsRequestHandler* | CourtDataManagementService.jar | 소송관계자의 입력 요청을 처리 |
| *SystemManagerRequestHandler.jar* | *SystemManagerRequestHandler* | CourtHealthMonitoring.jar  CourtInfoManagementService.jar | 시스템 관리자 SW의 입력 요청을 처리 |
| *CourtServiceControlService.jar* | *CourtServiceControlService* | OnlineCourtService.jar  CourtBroadcastingService.jar | 디지털 법정 연동 서비스 컨트롤 |
| *ScreenControlService.jar* | *ScreenControlService* | - | 디지털 법정 현장 공용 스크린 컨트롤 |
| *InMemoryCachingService.jar* | *InMemoryCachingService* | CourtDataHandler.jar | 법정 자료 DB를 캐쉬 하여 관리하며 Preview data 생성 |
| *MultimediaDeviceControlService.jar* | *MultimediaDeviceControlService* | - | 디지털 법정 현장 멀티미디어 입력 장비 컨트롤 |
| *OnlineCourtService.jar* | *OnlineCourtService* | *CourtDispatcherService.jar* | 영상 법정 시스템 연동 컨트롤 |
| *CourtBroadcastingService.jar* | *CourtBroadcastingService* | *CourtDispatcherService.jar* | 재판 방송 시스템 연동 컨트롤 |
| *CourtDispatcherService.jar* | *CourtDispatcherService* | - | 외부 시스템 인터페이스 컨트롤 |
| *CourtHealthMonitoring.jar* | *CourtHealthMonitoring* | *SystemLogDBHandler.jar* | 디지털 법정 시스템 운영 현황 모니터링 |
| *SystemLogDBHandler.jar* | *SystemLogDBHandler* | - | 디지털 법정 시스템 운영 Log 관리 |

# Component Design Description

**<작성 방법>**

* 아키텍처적으로 중요한 Component **3개**를 선정합니다.
* 5.3의 Sequence Diagram 5개 중 3개를 선택하고, 각 SD를 구성하는 Component 중 하나를 각각 선정합니다.
* 각 Component의 내부를 Class Diagram으로 모델링하고, Component의 대표적인 내부 Behavior를 Class 수준에서 UML Sequence Diagram으로 기술합니다.
* 해당 Component의 Full Specification이 아직 제공되지 않으니 (UC나 User Story 형태), 상위 수준의 Use Case Model (3.1)의 시나리오를 참고해서, 대표 시나리오를 하나 고안해야 합니다. (UC 형태로 정리할 필요는 없습니다.)
* 5.3 Behavior Model에서 정의한 Sequence Diagram과 연동해서 정의합니다.
* Use Case Model (3.1)의 시나리오와 연동됩니다.

## *Component 1* Design Description

### Overview

**<작성 방법>**

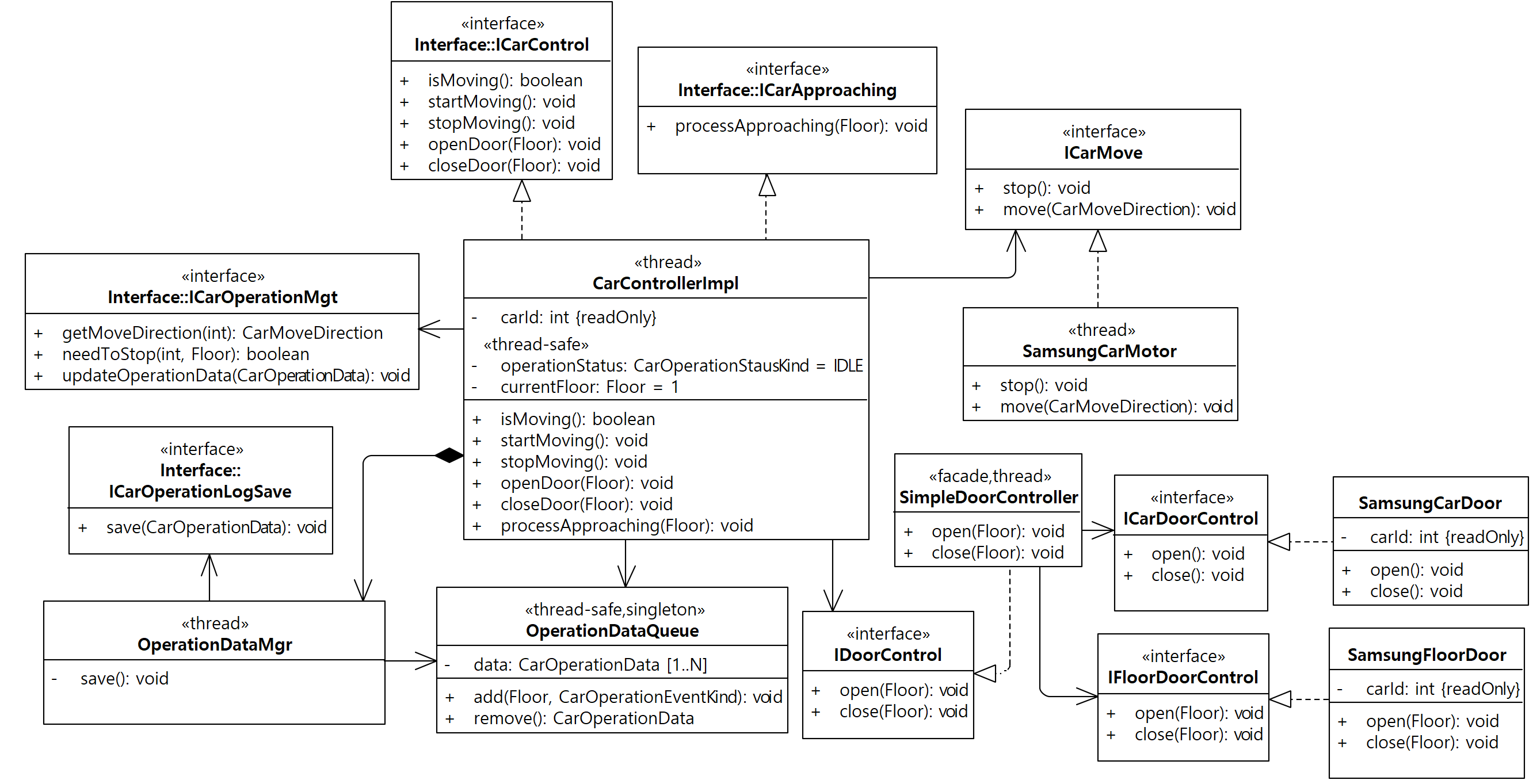
* Component를 이해하기 쉽게 설명합니다.
* Component가 지니는 기능 요구사항 (Primary Functionality)과 QA/QAS 측면에서 친절하게 설명합니다.



### Component Structure Diagram

<작성 방법>

* Component에 주어진 기능 및 QA/QAS를 충족시키기 위하여 필요한 Detailed Design을 수행하고 UML Class Diagram으로 명세합니다.
* Component를 구현하기 위하여 필요한 Class를 모두 식별합니다.
* 기 정의된 Interface를 C&C 하게 활용하고 정의합니다.
* UML Class Diagram의 Syntax & Semantics를 정확하게 따릅니다.
* 각종 Relationships, Visibility, Multiplicity를 정확하게 사용합니다.
* 객체지향 Design Principles를 적절하게 적용합니다.
* SOLID
* GoF Design Patterns
* Class Diagram에 대한 전반적인 설명을 포함합니다.
* 적용된 Design Principle 과 Design Pattern에 대해 친절히 설명합니다.



### Element List

**<작성 방법>**

* 위의 Class diagram에 표현된 모든 요소(Class, Interface)에 대한 역할을 설명합니다.
* 각 요소가 Component에 부여된 요구사항(기능, 품질, 제약사항 등)에 어떻게 기여하는 지에 대해서도 추가적으로 설명합니다.

|  |  |
| --- | --- |
| **Name** | **Responsibility** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

### Design Rationale

**<작성 방법>**

* Component 내부 설계를 통해서 어떤 QA가 어떻게 구현/달성되는지 논리적으로 설명합니다.
* 이 Component와 관련된 QA에 대하여 이 QA 달성을 위하여 직접적으로 기여하는 design elements(class 및 interface)를 나열합니다.
* 이들 관련된 design elements들이 QA 달성 측면에서 최적의 Detailed Design 인지에 대하여 정당화를 시도합니다.
* 적용된 Pattern 및 Tactic이 있다면 해당 Pattern/Tactic을 구체적으로 제시한다.
* 고려될 수 있는 다른 설계 후보가 있었다면 그 설계 후보와 비교하여 정당화한다.
* 단, 적용 가능한 부분에 대해서만 작성합니다. 즉, 모든 QA가 Detailed Component Design에 영향을 주지는 않습니다.

|  |  |  |
| --- | --- | --- |
| **QA/QAS** | **Relevant Elements** | **Description** |
| QA1 |  |  |
| QAS-04 | - | - |
| … |  |  |

### Component Behavior Diagram

**<작성 방법>**

* Component를 정의한 Class Diagram을 기준으로, 이 Component의 대표적인 내부 Behavior를 Class/Object Instances 수준에서 UML Sequence Diagram으로 모델링하고 친절하게 설명합니다.
* 5.3 Behavior Model에서 정의한 Sequence Diagram과 연동해서 정의합니다.
* Use Case Model (3.1)의 시나리오와 연동됩니다.
* 시작은 Component의 Provided Interface를 구성하는 한 Operation이 외부로부터 호출되는 것으로 시작됩니다.

스크린샷, 직사각형이(가) 표시된 사진

자동 생성된 설명

## *Component 2* Design Description

## *Component 3* Design Description

# Architecture Traceability Summary

## Architecture Traceability Graph

**<작성 방법>**

* Stakeholder로부터 시작해서 Detailed Component Design까지 이르는 Full Traceability (추적성)을 한 눈에 파악 가능한 그래프/그림/표를 작성합니다.
* 각 item의 양방향 추적성을 모두 확보합니다.
* 각 Design Decision 관련 item들의 양방향 추적성을 모두 확보합니다.

텍스트, 스크린샷, 폰트, 그래픽이(가) 표시된 사진

자동 생성된 설명

## Summary of Traceability Items

**<작성 방법>**

* Architecture Traceability Graph에서 사용한 모든 Item을 설명하는 Summary입니다.

|  |  |  |
| --- | --- | --- |
| **Traceability Item** | | **Description** |
| **ID** | **Title** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**APPENDIX**

1. **사용한 Tactics 설명 (1페이지씩)**
2. **사용한 Reference Architecture (1페이지씩)**
3. **사용한 Architecture/Deployment Styles/Patterns (1페이지씩)**
4. **사용한 Design Patterns (1페이지씩)**