네, 제공해주신 '소프트웨어 아키텍처 명세' 문서의 각 장에 명시된 '[작성 방법]'과 '[점검사항]'을 평가 기준 문서 형식으로 정리해 드리겠습니다.

아래 내용은 각 장과 절의 순서에 따라 평가 기준이 될 수 있는 작성법과 점검사항을 정리한 것입니다.

**소프트웨어 아키텍처 명세 평가 기준**

**1. Business Drivers**

* **1.1.1. Stakeholders List (이해관계자 목록)**
  + **작성 방법** 1
    - 제품의 획득, 개발, 인증, 배포/판매, 운영/관리, 유지보수 등 다양한 단계의 이해관계자(조직)를 식별해야 함2.
    - 각 이해관계자의 역할, 관심사, 애로사항 등의 배경을 기술해야 함3.
  + **점검 사항**
    - (해당 항목에 대한 명시적인 점검사항 없음)
* **1.1.2. Business Goals (비즈니스 목표)**
  + **작성 방법** 4
    - 이해관계자의 관심사항을 반영하여, 조직이 달성하고자 하는 구체적이고 측정 가능한 비즈니스 가치를 정의해야 함5. (예: 시장점유율 확대, 고객 만족도 개선 등) 6
    - 비즈니스 목표와 아키텍처와의 연관성을 기술해야 함7.
    - 조직의 전략적 방향과 현재 상황에 따라 비즈니스 목표의 중요도를 결정해야 함8.
  + **점검 사항** 9
    - 비즈니스 목표가 구체적이고 측정 가능한 비즈니스 가치를 반영하였는가? 10
    - 비즈니스 목표의 중요도에 대한 판단 근거가 합당한가? 11
* **1.2. Business Constraints (비즈니스 제약사항)**
  + **작성 방법** 12
    - 예산, 시간, 법적, 자원 제약 등을 포함하여 구체적이고 명확하게 기술해야 함13.
    - 비즈니스 제약사항이 아키텍처에 미치는 영향을 기술해야 함14.
  + **점검 사항** 15
    - 비즈니스 제약사항 서술이 구체적이고 명확한가? 16
    - 비즈니스 제약사항의 아키텍처에 대한 영향도 서술이 타당한가? 17

**2. System Context**

* **2.1. System Context Diagram (시스템 컨텍스트 다이어그램)**
  + **작성 방법** 18
    - 개발 대상 시스템과 외부 개체(External Entity)를 명확하게 구분해야 함19.
    - 시스템과 상호작용하는 외부 개체와의 인터페이스를 Provided/Required Interface 관점에서 표현해야 함20.
    - 사용자와 시스템 간의 사용자 인터페이스는 화면별로 별도의 인터페이스로 표시해야 함21.
    - 외부 개체가 다수일 경우, 이에 대한 표현과 함께 개수에 대한 가정 및 근거를 기술해야 함22.
  + **점검 사항** 23
    - 시스템의 경계(정의)가 명확한가? 24
    - 시스템과 직접 상호 작용하는 모든 대상(사용자, 시스템, 장치 등)이 외부 개체로 식별되었는가? 25
* **2.2. External Entity List (외부 개체 목록)**
  + **작성 방법** 26
    - 비즈니스 목표와 관련된 이해관계자 중 시스템과 직접 상호작용하는 대상을 모두 외부 개체로 도출해야 함27.
    - 이해관계자가 직접 상호작용하지 않고, 그들이 운영하는 시스템을 통해 상호작용한다면 해당 시스템을 외부 개체로 도출해야 함28.
    - 각 외부 개체의 시스템 관점에서의 역할 및 특성을 기술해야 함29.
    - 사용자 유형의 경우 사용 환경, 전문성, 기술 수준 등을, 그 외 시스템/장치 유형은 HW/SW 사양, 품질 수준(성능, 신뢰성 등)을 구체적으로 기술해야 함30.
  + **점검 사항** 31
    - 외부 개체의 역할과 특성이 명확하고 구체적으로 기술되었는가? 32
* **2.3. External Interface List (외부 인터페이스 목록)**
  + **작성 방법** 33
    - 시스템 컨텍스트 다이어그램의 모든 외부 인터페이스를 나열해야 함34.
    - 시스템 관점에서 각 외부 인터페이스의 역할을 기술해야 함35.
    - 사용자 인터페이스는 입출력 데이터를, 디바이스/시스템 인터페이스는 입출력 데이터, 프로토콜, 데이터 포맷을 기술해야 함36.
    - 품질(보안, 성능 등)에 영향을 미치는 특성(데이터 암호화, 인증 방식, 입출력 크기, 주기/빈도 등)을 기술해야 함37.
    - 인터페이스에서 발생 가능한 오류를 식별하고 오류 처리 방안을 명세해야 함38.
  + **점검 사항** 39
    - 시스템과 외부 개체와의 인터페이스가 명확하고 구체적으로 표현되었는가? 40

**3. Architectural Drivers**

* **3.1.1. Use Case Diagram (유스케이스 다이어그램)**
  + **작성 방법** 41
    - 아키텍처 설계에서 고려할 핵심 기능을 모두 유스케이스로 식별해야 함42.
    - 시스템과 상호작용하는 주요 대상을 액터(Actor)로 식별해야 함43.
    - 유스케이스의 이름은 액터에게 제공되는 결과물이나 서비스를 명확하게 기술해야 함44.
    - 관계(연관, 포함, 확장, 일반화)는 규칙에 맞게 사용해야 함45.
  + **점검 사항** 46
    - 유스케이스와 액터 사이, 유스케이스 간, 액터 간의 관계가 올바른가? 47
* **3.1.2. Actor List (액터 목록)**
  + **작성 방법** 48
    - 시스템 컨텍스트 다이어그램의 외부 개체를 모두 액터로 도출해야 함49.
    - 액터에 대한 설명은 시스템 관점에서의 역할을 파악할 수 있도록 구체적이어야 함50.
  + **점검 사항**
    - (해당 항목에 대한 명시적인 점검사항 없음)
* **3.1.3. Use Case List (유스케이스 목록)**
  + **작성 방법** 51
    - 다이어그램의 모든 유스케이스를 나열하고, 제공되는 기능을 명확하고 구체적으로 표현해야 함52.
    - Description에는 기능 개요, BV(Business Value) 설정 근거, AI(Architectural Impact) 설정 근거를 명확하게 기술해야 함53.
    - BV는 비즈니스 가치를, AI는 아키텍처 영향도를 평가함54.
    - Business Drivers는 해당 유스케이스가 어떤 비즈니스 목표/제약사항에서 도출되었는지 나타내야 함55.
  + **점검 사항** 56
    - 구조에 영향을 미치는 기능 명세가 충분한가? 57
    - 유스케이스들의 우선순위(BV, AI)에 대한 평가가 타당한가? 58
* **3.1.4. Use Case Description (유스케이스 기술서)**
  + **작성 방법**
    - **Pre-Condition**: Basic Flow가 시작되기 위해 필요한 시스템 상태를 기술함59.
    - **Post-Condition**: Basic Flow가 종료되었을 때 충족해야 할 시스템 상태를 기술함60.
    - **Basic Flow**: 시스템의 기본 기능 동작만 기술하며, 각 스텝은 액터 또는 시스템을 주어로 하는 능동태 문장으로 하나의 동작을 기술함61.
    - **Alternative Flow**: 추가 기능 및 예외 상황 처리 동작을 기술함62.
  + **점검 사항** 63
    - 시나리오 명세가 구체적이고 명확한가? 64
    - 아키텍처 측면에서 고려할 이슈를 포함하는 대안 시나리오가 식별되었는가? 65
    - 해당 시나리오에 관련된 모든 액터가 시나리오에서 언급되었는가? 66
* **3.2.1. QA Scenario List (품질 속성 시나리오 목록)**
  + **작성 방법** 67
    - 비즈니스 목표 달성에 필요한 모든 품질 속성(QA)을 식별해야 함68.
    - Description에는 품질 요구사항 개요, BV 설정 근거, AI 설정 근거를 명확하게 기술해야 함69.
    - Business Drivers는 해당 QA가 어떤 비즈니스 목표/제약사항에서 도출되었는지 나타내야 함70.
  + **점검 사항** 71
    - 비즈니스 목표를 실현하는데 필요한 품질 요구사항이 충분한가? 72
    - 품질 요구사항의 우선순위(BV, AI)에 대한 평가가 타당한가? 73
* **3.2.2. QA Scenario (품질 속성 시나리오)**
  + **작성 방법/점검 사항** 74
    - Description이 QA 시나리오에 대한 간략한 설명을 명확하게 기술하였는가? 75
    - Source 및 Stimulus가 구체적인가? (예: '기능 추가' (X), '새로운 스케줄링 방법 추가' (O)) 76
    - Artifact 및 Environment가 구체적인가? (평가 대상 시스템과 평가 환경을 명확히 판단 가능해야 함) 77
    - Response가 구체적인가? (평가 시 측정 대상이 되는 동작/결과를 구체적으로 기술해야 함) 78
    - Response Measure가 Response와 일치하며 구체적인가? (QA 충족 여부를 판단할 수 있도록 측정 유형과 수치가 구체적이어야 함) 79
* **3.3. Architectural Constraint (아키텍처 제약사항)**
  + **작성 방법** 80
    - 앞서 도출한 비즈니스 제약사항 중 아키텍처 설계 결정에 영향을 미칠 수 있는 요소를 모두 도출해야 함81.
  + **점검 사항** 82
    - Title이 제약사항의 핵심을 명확하게 표현하는가? 83
    - Description이 제약사항의 내용을 구체적이고 명확하게 기술하는가? 84
    - Business Drivers가 올바르게 연결되었는가? 85

**4. High Level Structure Design Description**

* **4.1.1. Conceptual Class List (개념 클래스 목록)**
  + **작성 방법** 86
    - 유스케이스별로 기능을 실현하는 데 필요한 개념 클래스 목록을 도출함 (반복적으로 정제)87.
    - 클래스 카테고리(entity, control, boundary 등)에 따라 스테레오타입 정보를 Type란에 기술함88.
  + **점검 사항** 89
    - 유스케이스 시나리오를 실현하기 위해 필요한 개념적 클래스가 충분히 도출되었는가? 90
    - 구현 수준의 클래스를 포함하고 있지 않은가? 91
* **4.1.2. Dynamic View (동적 뷰)**
  + **작성 방법** 92
    - UML Communication Diagram을 이용해 유스케이스 시나리오를 개념 클래스 인스턴스 간의 메시지 흐름으로 표현해야 함93.
    - 메시지 레이블은 'ReturnData := MessageName(argument-list)' 형식으로 표현해야 함94.
  + **점검 사항** 95
    - 인스턴스 간 메시지가 형식에 맞춰 작성되었는가? 96
    - 메시지 상호작용이 구체적이고 명확한가? 97
    - 유스케이스 시나리오와 일관성이 있는가? 98
* **4.1.3. Static View (정적 뷰)**
  + **작성 방법** 99
    - 도출된 개념 클래스 간의 관계(association)를 추가하여 클래스 다이어그램을 작성해야 함100.
    - 관계(Association)에 다중성(multiplicity) 정보를 추가해야 함101.
    - Entity 클래스에는 속성(attribute)을 추가해야 함102.
  + **점검 사항** 103
    - 정적 뷰의 클래스 다이어그램과 동적 뷰의 커뮤니케이션 다이어그램 간에 일관성이 있는가? 104
* **4.2. Quality Driven Architectural Design (품질 기반 아키텍처 설계)**
  + **Quality Driven Architectural Decision Structure (설계 결정 구조)**
    - **작성 방법**: QA를 만족시키기 위한 설계 이슈(Design Issue)와 설계 옵션(Design Option) 간의 의사결정 구조를 도식화해야 함105105105105105.
    - **점검 사항**: 품질 요구사항에 대한 아키텍처 설계 결정 구조가 적절한가? 106
  + **Design Decisions (설계 결정)**
    - **작성 방법**: 이슈의 배경/가정, 이슈 정의를 명확하게 기술해야 함107107107107107.
    - **점검 사항**: 설계 이슈가 다루는 문제가 구체적이고 명확한가? 108
  + **Design Options (설계 옵션)**
    - **작성 방법**: 관련된 모든 설계 옵션을 기술하고, 각 옵션에 대한 설명은 해당 솔루션(패턴, 택틱 등)을 시스템에 적용할 때의 구체적인 설계 결정을 제시해야 함109109109109109109109109109109.
    - **점검 사항**: 설계 옵션에 대한 설명이 구체적인가? 110
  + **Decision and Rationale (결정 및 근거)**
    - **작성 방법**: 각 설계 옵션의 장단점을 QA 및 관련 관심사 측면에서 구체적으로 제시하고, 최종 선택에 대한 근거를 기술해야 함111.
    - **점검 사항**: 설계 옵션에 대한 평가와 선택 근거가 타당한가? 112
* **4.3. Component & Connector View (컴포넌트 & 커넥터 뷰)**
  + **작성 방법**: 4.2절의 아키텍처 설계 결정 중 C&C 뷰에 반영된 부분을 어떻게 표현했는지 기술해야 함113113113113.
  + **점검 사항**: 아키텍처 설계 결정이 구체적으로 올바로 반영되어 있는가? 114
  + **Component List (컴포넌트 목록)**
    - **작성 방법**: 컴포넌트의 역할, 종류(Kind), 기능/품질 요구사항, 관련 아키텍처 드라이버(ADs)를 기술해야 함115115115115115115115115115.
    - **점검 사항**: Property Description이 구체적이고 관련 ADs와 일관성이 있는가? 116
  + **Connector List (커넥터 목록)**
    - **작성 방법**: 커넥터의 상호작용 역할, 프로토콜, 데이터 포맷이 어떤 품질/기능과 관련 있는지, 관련 ADs는 무엇인지 기술해야 함117117117117.
    - **점검 사항**: Property Description이 구체적이고 관련 ADs와 일관성이 있는가? 118
* **4.4. Deployment View (배포 뷰)**
  + **작성 방법**: 4.2절의 아키텍처 설계 결정 중 배포 뷰에 반영된 부분을 어떻게 표현했는지 기술해야 함119119119119.
  + **점검 사항**: 아키텍처 설계 결정이 구체적으로 배포 뷰에 올바로 반영되어 있는가? 120
  + **Deployment Diagram (배포 다이어그램) 및 명세**
    - **작성 방법**: 시스템 운용에 필요한 노드, 통신 경로, 실행 환경(플랫폼)을 표현하고 각 요소의 역할과 특성, 결정 근거를 ADs와 연관지어 구체적으로 기술해야 함121121121121121121121121121121121121121121121121121121121121121121121121121.
    - **점검 사항**: 각 노드, 실행 환경, 통신 경로에 대한 설명이 구체적인가? 122122122122122122122122122
  + **Artifact Definition Diagram (아티팩트 정의 다이어그램) 및 명세**
    - **작성 방법**: 노드에 설치될 파일들을 아티팩트(Artifact)로 식별하고, 각 아티팩트가 어떤 컴포넌트로 구성되는지 기술해야 함123123123123.
    - **점검 사항**: C&C 뷰의 각 컴포넌트가 1개 이상의 아티팩트에 포함되었는가? 124각 아티팩트의 기능/역할이 명확히 설명되었는가? 125

**5. Detailed Component Design Description**

* **5.1.1. Overview (개요)**
  + **작성 방법**: 컴포넌트의 역할, 부여된 기능 요구사항, 만족해야 할 품질 요구사항 및 제약사항을 구체적으로 서술해야 함126.
  + **점검 사항**: 컴포넌트의 요구사항이 명확하게 정의되었는가? 127
* **5.1.2. Quality Driven Component Design (품질 기반 컴포넌트 설계)**
  + **작성 방법**: 상위 수준 설계(4.2절)와 동일하게 설계 결정 구조, 설계 이슈, 설계 옵션, 최종 결정 및 근거를 컴포넌트 내부 설계 관점에서 구체적으로 기술해야 함128128128128128128128128128128128128128128128128128128128128128128128128128.
  + **점검 사항**: 품질 요구사항에 대한 컴포넌트 설계 결정 구조가 적절한가? 129설계 이슈와 옵션 설명이 구체적인가? 130옵션 평가와 선택 근거가 타당한가? 131
* **5.1.3. Module View (모듈 뷰)**
  + **작성 방법**: 5.1.2절의 설계 결정이 모듈 뷰에 어떻게 반영되었는지 기술해야 함132132132132.
  + **점검 사항**: 컴포넌트 설계 결정이 모듈 뷰에 구체적으로 반영되어 있는가? 133
  + **Class Diagram (클래스 다이어그램)**
    - **작성 방법**: 컴포넌트 구현에 필요한 모든 클래스와 인터페이스를 식별하고, 관계, 속성, 오퍼레이션을 구체적으로 명시하며, 필요시 패키징을 해야 함134.
    - **점검 사항**: 클래스 다이어그램이 구체적이고 올바로 작성되어 있는가? 135
  + **Element List (요소 목록)**
    - **작성 방법**: 다이어그램의 모든 요소(패키지, 클래스, 인터페이스)의 역할을 설명하고, 컴포넌트 요구사항에 어떻게 기여하는지 설명해야 함136.
    - **점검 사항**: 요소의 설명이 구체적인가? 137
  + **Sequence Diagram (시퀀스 다이어그램)**
    - **작성 방법**: 컴포넌트 인터페이스의 각 오퍼레이션이 내부 모듈 인스턴스 간의 상호작용으로 어떻게 실현되는지 기술해야 함138.
    - **점검 사항**: 시퀀스 다이어그램이 구체적이고 올바로 작성되어 있는가? 139

**6. Architectural Evaluation**

* **6.1. Architectural Evaluation (아키텍처 평가)**
  + **작성 방법**: 제시된 양식에 맞춰 QA 시나리오와 아키텍처 결정에 대한 분석(위험요소, 민감점, 절충점, 비위험요소)을 작성해야 함140.
  + **점검 사항**: 아키텍처 결정의 민감점, 절충점, 위험요소, 비위험요소 분석이 타당한가? 141아키텍처 결정에 대한 추론(Reasoning)이 타당한가? 142

**7. Appendix**

* **7.1. Detailed Component Specification (상세 컴포넌트 명세)**
  + **점검 사항**: 해당 컴포넌트의 모든 인터페이스(Provided/Required)가 정의되었고, 역할/기능 설명이 명확하며 다이어그램과 부합하는가? 143
* **7.2. Interface specifications (인터페이스 명세)**
  + **작성 방법**: 인터페이스를 구성하는 각 오퍼레이션(operation)별로 구체적인 인자 및 반환 타입을 기술해야 함144.
  + **점검 사항**: 컴포넌트에 할당된 모든 기능이 오퍼레이션으로 할당되었는가? 145오퍼레이션 이름이 기능을 명확히 나타내는가? 146인자/반환 타입이 올바르게 정의되었는가? 147