

클라우드 네이티브 전환 사전 진단(CNTA)

컨테이너 우선 원칙과 본사업 전 전환 범위·공수 확정 절차

CNTA(Cloud Native Transformation Assessment, 클라우드 네이티브 전환 사전 진단)는 본사업을 발주하기 전에 대상 애플리케이션을 분석해, 컨테이너로 전환할 수 있는지와 전환에 드는 공수를 수치로 제시하는 절차입니다.

목차

클라우드 네이티브 전환 사전 진단(CNTA): 컨테이너 우선 원칙과 본사업 전 전환 범위·공수 확정 절차

- 1장. 전환 사업의 견적이 들어지는 지점과 사전 진단이 필요한 이유
 - 1.1 배치 위치만 정한 전환 계획에서 범위와 비용이 어긋나는 구조
 - 1.2 본사업 발주 전에 확정할 두 가지: 가상화 예외 목록과 컨테이너 전환 공수
 - 1.3 2030년까지 이어지는 공공 전환 일정과 순차 전환의 근거
- 2장. 사전 진단 CNTA의 정의와 절차: 다섯 단계와 단계별 산출물
 - 2.1 CNTA(Cloud Native Transformation Assessment)가 답하는 여섯 가지 질문
 - 2.2 사전 조치에서 결과 공유까지 다섯 단계의 활동과 산출물
 - 2.3 3주 진단 일정의 구성과 투입 역할, 고객 기관의 사전 준비물
 - 2.4 클라우드 사업자의 준비도 평가와 코드 기반 사전 진단의 차이
- 3장. 진단이 재는 것: 체크리스트 13개 분류와 정적 분석
 - 3.1 고객 기관이 작성하는 현황 체크리스트 13개 대분류의 구성
 - 3.2 소스와 바이너리를 규칙으로 검사하는 정적 분석의 산출 지표
 - 3.3 체크리스트 조사와 정적 분석 결과를 합쳐 전환 가능 여부를 판정하는 절차
- 4장. 난이도와 공수 산정: Story Point에서 일수까지
 - 4.1 이슈 한 건마다 0점에서 13점까지 매기는 Story Point 여섯 등급
 - 4.2 Story Point를 작업 일수로 바꾸는 환산 관행과 그 한계
 - 4.3 익명 진단 사례 6건에서 확인되는 점수와 일수의 관계
 - 4.4 Story Point와 공공 소프트웨어 사업 대가 산정을 잇는 세 경로
- 5장. 전환과 현대화를 함께 해야 하는 이유: 지원 종료 일정과 JVM 컨테이너 인식
 - 5.1 운영체제에서 프레임워크까지 네 계층의 지원 종료 일정
 - 5.2 JVM이 컨테이너 자원 한도를 인식하기 시작한 JDK 버전
 - 5.3 상용 WAS 종속 코드가 필수 수정 대상이 되는 근거
 - 5.4 순수 Java와 플랫폼 종속 네이티브 코드의 Unix→Linux 전환 난이도 차이
- 6장. 전환 원칙: 컨테이너 전환이 기본, 가상화 전환은 예외
 - 6.1 모든 대상에 컨테이너 전환을 먼저 검토하는 세 가지 원칙
 - 6.2 컨테이너 전환과 VM 이전이 남기는 기술 부채 비교
 - 6.3 리눅스 실행 가능성으로 정하는 컨테이너 전환 판정 기준
 - 6.4 가상화 전환을 허용하는 예외 사유와 기록 항목
 - 6.5 예외로 남은 VM을 컨테이너 플랫폼에서 함께 운영하는 구성
- 7장. 정부 상세설계·본사업·NIA 가이드가 비워 둔 자리
 - 7.1 상세설계 사업 제안요청서가 사업자에게 넘기는 판단 항목
 - 7.2 상세설계 대상 시스템에 반복해 나타나는 구버전 스택과 Java 외 언어
 - 7.3 NIA 안내서와 가이드에 없는 정량 기준 일곱 가지
 - 7.4 본사업 제안요청서를 기준으로 사전 진단에서 미리 확정할 여덟 항목
 - 7.5 대상 지정 이전 단계를 맡는 사전 진단과 상세설계의 역할 분담

8장. 진단 이후: 우선순위, 파일럿, 단계적 확산

8.1 업무 중요도와 Mandatory 점수로 나누는 전환 우선순위 4분면

8.2 상태가 없고 기동이 빠른 서비스를 고르는 파일럿 선정 기준

8.3 컨테이너 전환을 먼저 완료하고 MSA를 뒤에 두는 본사업 분할

8.4 사전 진단 수행 결정과 진단 대상 시스템·범위 선정

부록: 참고문헌과 용어 정의

Appendix A — 참고문헌 (References)

Appendix B — 용어 정의 (Glossary)

클라우드 네이티브 전환 사전 진단(CNTA): 컨테이너 우선 원칙과 본사업 전 전환 범위·공수 확정 절차

클라우드 네이티브(Cloud Native)는 클라우드의 장점을 최대한 활용해 정보시스템을 구축하고 실행하는 환경을 가리키며[S43], 공공부문에서는 기존 시스템을 이 환경으로 옮기는 전환 사업이 2024년 이후 계속 발주되고 있습니다. CNTA(Cloud Native Transformation Assessment, 클라우드 네이티브 전환 사전 진단)는 본 사업을 발주하기 전에 대상 애플리케이션을 분석해, 컨테이너로 전환할 수 있는지와 전환에 드는 공수를 수치로 제시하는 절차입니다. 결론을 먼저 적으면 전환의 기본은 컨테이너 전환이고 가상화(VM) 전환은 사유가 확인된 예외이며, 사전 진단은 그 예외 목록과 컨테이너 전환 공수를 본사업 전에 확정하는 절차입니다. 이 백서는 그 근거를 여덟 장에 나누어 제시합니다. 1장은 건적이 어긋나는 원인을, 2·3·4장은 진단의 절차와 측정 항목, 공수 산정을, 5·6장은 현대화를 함께 해야 하는 이유와 컨테이너 우선 원칙을, 7·8장은 공공 발주 문서가 비워 둔 기준과 진단 이후의 추진 순서를 다룹니다.

1장. 전환 사업의 건적이 틀어지는 지점과 사전 진단이 필요한 이유

사전 진단이 필요한 이유는 공공 발주 문서의 조항에서 확인할 수 있습니다. 2025년에 공고된 한 지방교육청의 전환 본사업 제안요청서는 서버 38식과 소프트웨어 132식을 품명으로만 신고 이기종 전환을 요구하며, 문서에 명시하지 않은 요구도 계약금액 범위 안에서 수용하도록 정하고 있습니다[S41]. 따라서 애플리케이션을 얼마나 고쳐야 하는지 모르는 상태로 발주하면, 그 차이는 본사업의 재작업 비용과 일정 지연으로 나타납니다. 이 구조를 발주 문서의 조항으로 확인하면 발주 전에 내릴 결정 두 가지가 나오고, 공공 전환 일정이 그 결정의 시기를 정합니다.

1.1 배치 위치만 정한 전환 계획에서 범위와 비용이 어긋나는 구조

공공기관의 정보화전략계획(ISP, Information Strategy Planning)은 전환 대상을 어느 클라우드에 둘 것인지를 기준으로 예산을 편성해 왔습니다. 배치 위치를 기준으로 삼으면 서버 이전비와 이용료는 계산할 수 있지만, 애플리케이션 코드와 미들웨어, 데이터베이스를 얼마나 고쳐야 하는지는 계산에 들어가지 않습니다. 수정 범위는 본사업 계약 뒤에야 드러나고, 제안요청서와 계약특수조건의 일반 조항에 따라 사업자의 보완·재작업 부담과 발주자의 일정 지연으로 이어집니다[S41][S42].

1.1.1 「어느 클라우드에 둘 것인가」로 편성한 예산에서 빠지는 애플리케이션 수정 비용

위치 중심 계획의 예산 항목

위치 중심 계획은 전환 대상 시스템을 어느 클라우드 서비스와 어느 센터에 배치할지를 먼저 정하고, 그 결과에 맞추어 예산을 편성합니다. 배치 위치를 정한 뒤에는 서버 규모에 따른 인프라 전환비와 클라우드 이용료를 중심으로 예산을 계산합니다. 이 계산에 필요한 입력값은 서버 대수와 CPU·메모리 용량이므로 자산 대장만으로도 금액을 산출할 수 있습니다. 반면 애플리케이션이 어떤 Java 버전과 WAS(Web Application Server, 웹 애플리케이션 서버)에 종속되어 있는지는 자산 대장이 다루지 않는 정보이므로 예산 항목에서 빠집니다.

계획 단계에 빠지는 수정 비용

계획 단계에 빠지는 비용은 대부분 애플리케이션 계층에서 발생하며, 코드 수정과 미들웨어 교체, 데이터 변환 세 가지로 나눌 수 있습니다. 예를 들어 NIA(한국지능정보사회진흥원)가 2025년 5월에 발간한 「클라우드 네이티브 구축·운영 가이드」의 설계기준 예시는 JDK 버전 변경에 따른 프레임워크 버전업과 이기종 DB의 쿼리·프로시저 수정을 모두 난이도 상으로 분류하고, 서버당 17 M/D(Man/Day, 한 사람의 하루 작업량)를 제시합니다 [S45]. 이 예시는 가상 서버로 옮기는 IaaS(Infrastructure as a Service) 전환을 기준으로 삼았는데도 애플리케이션 수정이 가장 무거운 항목에 속합니다. 위치만 정한 계획과 사전 진단을 거친 계획은 아래 표처럼 예산 항목의 수부터 다릅니다.

예산 항목	위치 중심 계획	사전 진단 기반 계획
인프라 전환비	현행 서버 대수 기준으로 산정	전환 후 구성 기준으로 산정
클라우드 이용료	현행 용량 기준으로 산정	컨테이너 자원 산정 결과로 산정
코드 수정비	항목 없음	이슈별 난이도 점수와 작업 일수
미들웨어 교체비	항목 없음	WAS 종속 코드의 수정 범위
데이터 변환비	항목 없음	이기종 DBMS 전환 대상과 수정 범위
가상화 예외 대상	구분 없음	사유와 재검토 시점을 기록한 목록

발주 문서의 비용 산정 항목

정부가 발주하는 상세설계 사업도 비용을 세 항목으로 나누어 산정하도록 요구합니다. 2026년 클라우드 네이티브 전환 상세설계 사업(2차) 제안요청서는 소요예산을 애플리케이션 개발비, 클라우드 이용료, 인프라 전환비로 산출하게 하고, 애플리케이션 개발비의 근거가 되는 기능점수(Function Point)는 제3자 검증을 받도록 정하고 있습니다[S36]. 위치 중심 계획은 이 세 항목 가운데 뒤의 두 항목만 채운 상태이므로, 애플리케이션 수정 범위에 따라 금액이 달라지는 첫 항목을 비워 둔 채로 발주 절차를 시작하게 됩니다.

진단이 바꾸는 예산 산정의 입력값

사전 진단은 예산 산정의 질문을 「어느 클라우드에 둘 것인가」에서 「무엇을 남기고 어디를 바꿀 것인가」로 바꿉니다. 질문이 바뀌면 입력값도 서버 대수에서 애플리케이션 구조로 바뀌고, 총사업비와 위험도, 단계 계획도 그 입력값에 맞추어 다시 산정해야 합니다. NIA의 「클라우드 네이티브 정보시스템 구축 발주자 안내서」도 서문에서 단순한 기술 인프라 위주의 클라우드 도입보다 클라우드 네이티브 정보시스템 구축이 필요하다고 밝히고 있어 [S43], 인프라만 계산한 예산은 정부 안내서가 권고하는 방향과도 맞지 않습니다.

1.1.2 제안요청서에 품명만 적힌 소프트웨어의 버전·언어·JDK 정보 공백

본사업 제안요청서의 현행 자원 목록

2025년에 공고된 한 지방교육청의 클라우드 네이티브 전환 본사업은 예산 약 46억 원, 사업 기간 600일 규모이고 전환 대상은 3개 시스템입니다[S41]. 제안요청서의 현행 자원 목록에는 서버 38식과 소프트웨어 132식이 실려 있으며, 서버 가운데 Solaris가 5식, Windows가 9식입니다. 주력 소프트웨어는 웹 서버 WebtoB, WAS인 JEUS, DBMS인 Oracle입니다. 그런데 이 목록은 품명만 실고 있어서 제품 버전과 개발 언어, JDK(Java Development Kit) 버전을 문서에서 확인할 수 없습니다[S41].

목표 구성이 요구하는 이기종 전환

같은 제안요청서가 제시하는 목표 구성은 관리형 Kubernetes(컨테이너의 배치와 운영을 자동화하는 오픈소스 플랫폼) 위에서 실행되는 마이크로서비스 6개이고, OS는 Rocky Linux로 단일화하며, 소프트웨어는 국산·공개 제품을 중심으로 이기종 전환하도록 요구합니다[S41]. 이 목표를 달성하려면 Solaris와 Windows에서 실행되던 애플리케이션을 리눅스로 옮겨야 하고, 상용 WAS와 DBMS에 종속된 코드도 함께 수정해야 합니다. 수정 범위는 현행 JDK와 WAS의 버전에 따라 크게 달라지는데, 그 버전이 발주 문서에 없으므로 제안사는 수정 범위를 추정해서 가격을 정해야 합니다. 아래 표는 이 제안요청서에 실린 정보와 실리지 않은 정보를 나란히 정리한 것입니다.

구분	제안요청서에 실린 정보	실리지 않았거나 확정되지 않은 정보
서버	38식, Solaris 5식과 Windows 9식 포함	OS별 세부 버전
소프트웨어	132식의 품명, 주력은 WebtoB·JEUS·Oracle	제품별 버전
애플리케이션	3개 시스템, 기능점수 3,148FP와 1,752FP	개발 언어, JDK 버전
목표 구성	마이크로서비스 6개, Rocky Linux, 이기종 전환	대상별 수정 난이도와 공수
누리집 규모	누리집 수	130개와 133개, 1,042개와 1,060개로 다르게 표기

상세설계 단계에 있던 버전 정보

이 시스템은 2024년 클라우드 네이티브 전환 상세설계 4차 사업의 대상이었고, 그 제안요청서에는 버전 정보가 실려 있었습니다. JDK는 1.6·1.7·1.8이 혼재하고, Java 외에 ASP와 PHP 5로 작성된 부분이 있으며, WebtoB 4, JEUS 6·8, IIS 10, Oracle 11g·12c·19c, MySQL 5.1이 함께 운영되는 구성입니다[S40]. 버전 정보는 앞 단계 문서에 있었지만 본사업 제안요청서로는 넘어오지 않았고, 본사업의 설계 자료는 보안서약을 한 뒤 입회해서 열람해야 합니다[S41]. 그 결과 본사업 제안사는 JDK 1.6과 ASP가 섞인 구성이라는 사실을 공고문만으로는 알 수 없습니다.

발주 단계에서 확정되지 않는 규모

규모를 나타내는 수치도 발주 단계에서 하나로 정리되지 않은 경우가 있습니다. 같은 제안요청서 안에서 전환 대상 누리집 수가 한 곳에는 130개, 다른 곳에는 133개로 적혀 있고, 전체 누리집 수도 1,042개와 1,060개로 다르게 적혀 있습니다[S41]. 버전과 규모가 모두 불확실하면 사업자는 그 위험을 제안 가격에 반영하거나 사업 수행 중에 범위 재협의를 요청할 수밖에 없습니다. 발주 문서의 정보 공백이 사업자 선정 이후의 비용 문제로 이어지는 경로가 여기에 있습니다.

1.1.3 계약금액 변동 없이 사업자에게 부과되는 보완·재작업 조항

명시되지 않은 요구의 수용 조항

지방교육청 본사업 제안요청서의 일반 요구사항에는 범위가 늘어나도 계약금액은 바뀌지 않도록 한 조항이 있습니다. 요지는 제안요청서에 구체적으로 명시되지 않은 부분도 계약금액의 변동이 없는 범위에서는 주관기관과 전환대상기관의 요구에 응해야 한다는 것입니다[S41]. 제출된 상세설계 결과를 검증할 때 아키텍처, 호환성, 성능, 보안 등의 문제가 확인되면 사업자에게 보완과 재작업을 요구할 수 있고, 그 비용은 본사업에 포함한다는 조항도 같은 문서에 있습니다[S41]. 두 조항을 함께 적용하면 계획 단계에서 파악하지 못한 수정 범위는 추가 예산 없이 사업자가 처리해야 하는 작업이 됩니다.

지체상금과 정보 제공 의무

행정·공공기관 클라우드 네이티브 전환지원 사업의 계약특수조건은 이행책임을 전환 사업자와 설계 사업자에게 우선 부여하고, 검사 미비로 사업 기간이 늘어나면 지체상금을 부과하도록 정하고 있습니다[S42]. 반면 전환대상기관에도 의무가 있습니다. 같은 계약특수조건은 전환대상기관이 OS, 웹 서버와 WAS, DBMS의 전환 사항과 데이터 이관, 보안정책 정보를 제공하고 클라우드 네이티브 전환 체크리스트를 최종 확인하도록 정하고 있습니다[S42]. 현행 시스템의 정보를 정리하는 일은 계약상으로도 발주 측이 맡은 책임입니다.

조항 요지	발생할 수 있는 영향
명시되지 않은 요구도 계약금액 범위 안에서 수용	사전에 파악하지 못한 수정 범위가 사업자의 추가 작업으로 전가
상세설계 결과 검증에 따른 보완·재작업 비용을 본사업에 포함	설계 단계의 누락이 본사업 원가 상승으로 연결
검사 미비로 기간이 늘면 지체상금 부과	재작업이 길어질수록 사업자의 손실과 개통 지연이 함께 증가
전환 전후 동등 수준의 성능 보장	전환 전 성능 기준이 없으면 검사 단계에서 판정 기준을 두고 이견 발생

표의 앞 두 조항과 성능 보장 조항은 지방교육청 본사업 제안요청서에, 지체상금 조항은 계약특수조건에 있습니다[S41][S42].

발주자에게 돌아오는 영향

이 조항들은 비용을 사업자에게 부과하지만 그 영향은 발주자에게도 돌아옵니다. 재작업이 늘면 개통 일정이 늦어지고, 사업자가 위험을 미리 가격에 반영하면 발주자는 필요한 금액보다 높은 가격으로 계약할 수 있습니다. 앞 단계의 산정 결과를 다시 계산한 사례도 발주 문서에 남아 있습니다. 2024년 상세설계 4차 사업에는 기존 컨설팅을 받은 4개 기관의 소요예산을 검증하고 재산출하는 과업이 들어 있고, 3차 사업은 2022년 심층 컨설팅 결과를 환경 변화를 고려해 재검토하도록 했습니다[S39][S40]. 따라서 범위와 비용이 어긋나는 원인은 사업자의 역량보다 계획 단계에서 애플리케이션 정보를 확보하지 못한 데 있습니다.

1.2 본사업 발주 전에 확정할 두 가지: 가상화 예외 목록과 컨테이너 전환 공수

CNTA의 산출물은 두 가지로 압축됩니다. 하나는 컨테이너로 전환할 수 없음이 확인된 대상과 그 사유를 적은 가상화 예외 목록이고, 다른 하나는 나머지 대상을 컨테이너로 전환하는 데 드는 공수입니다. 정부의 상세설계 제안요청서는 전환유형을 구분하는 기준을 「전환시 난이도, 수정의 범위」로만 적고 난이도 등급표와 대상 선정

기준은 사업자의 과업으로 넘기고 있습니다[S36]. 이 두 가지는 발주자가 사업자를 선정하기 전에 직접 확보해 두어야 하는 정보입니다.

1.2.1 컨테이너 전환을 기본으로, 가상화 전환을 사유가 확인된 예외로 두는 원칙

컨테이너 전환 우선 검토 원칙

모든 전환 대상은 컨테이너 전환을 먼저 검토하고, 가상화 전환은 컨테이너로 전환할 수 없음이 확인된 경우에만 허용합니다. 컨테이너는 애플리케이션과 실행 환경을 하나의 이미지로 묶어 어느 서버에서나 같은 방식으로 실행하는 기술이며[S01], 클라우드 네이티브 환경의 배포·확장 기능은 이 단위를 전제로 동작합니다. 가상 서버로 만 옮긴 시스템은 위치가 바뀌었을 뿐 코드와 구조의 문제를 그대로 가지고 있습니다. 담당자의 편의, 촉박한 일정, 기존 방식에 대한 익숙함은 가상화 전환의 사유로 인정하지 않습니다.

정부 안내서의 도입 순서와 성숙도 단계

이 원칙은 정부 안내서가 제시하는 순서와 같습니다. NIA의 발주자 안내서는 클라우드 네이티브 구성요소의 도입 순서를 컨테이너, 데브옵스와 CI/CD(지속적 통합·배포), MSA(Microservices Architecture, 마이크로서비스 아키텍처) 순으로 제시하고, 적합성 검토 점수가 낮아 MSA를 이루는 경우에도 컨테이너는 부분 도입 대상으로 둡니다[S43]. 성숙도 모델에서도 IaaS 전환은 가장 낮은 전환 단계에 해당합니다. 2021년 안내서는 Level 1을 「클라우드 준비단계」로, 2025년 가이드는 「클라우드 시작 단계」로 정의하고 있어[S43][S45], 가상 서버로 옮기는 것만으로는 정부 문서가 말하는 클라우드 네이티브 단계에 도달하지 못합니다.

설계에 포함된 가상화 예외

이 원칙은 가상화를 금지하지 않으며, 예외를 처음부터 설계에 포함합니다. 리눅스나 Windows 컨테이너로 옮길 수 없는 Windows 전용 소프트웨어, 커널이나 특수 하드웨어에 직접 의존하는 시스템, 소스나 제작사가 없는 상용 패키지는 컨테이너로 전환하기 어려운 대표적인 경우입니다. 다만 예외로 분류한 대상에는 사유와 해소 조건, 재검토 시점을 함께 기록해서 가상화 상태가 최종 구성으로 굳어지지 않게 합니다. 예외 사유의 세부 판정 기준과 예외 대상을 수용하는 방법은 6장에 정리되어 있습니다.

구분	전환 방식	적용 조건
기본	컨테이너 전환	모든 전환 대상에 먼저 적용하며 별도 사유가 필요하지 않음
예외	가상화(VM) 전환	컨테이너로 전환할 수 없음이 확인되고, 사유·해소 조건·재검토 시점을 기록한 대상

원칙이 고정하는 진단의 질문

원칙을 먼저 정해야 하는 이유는 진단의 질문이 원칙에 따라 달라지기 때문입니다. 원칙이 없으면 진단은 가상화와 컨테이너를 대상마다 비교하는 작업이 되고, 그 비교에는 일정과 익숙함 같은 요소가 다시 개입합니다. 반면 컨테이너 전환을 기본으로 정해 두면 진단의 질문은 「컨테이너로 전환하는 데 무엇이 제약이 되고, 그 제약을 해소하는 데 얼마가 드는가」 하나로 고정됩니다. 이렇게 질문을 고정해야 산출물도 예외 목록과 전환 공수 두 가지로 정리됩니다.

1.2.2 상세설계·본사업 발주에 앞선 사전 진단 수행 결정과 대상 선정

발주 전에 내릴 결정 두 가지

의사결정권자가 내릴 결정은 두 가지입니다. 첫째, 상세설계나 본사업을 발주하기 전에 대상 시스템의 사전 진단을 먼저 수행하기로 정합니다. 둘째, 진단 대상 시스템과 진단 범위를 고릅니다. 대상을 고르는 기준과 진단 이후의 추진 순서는 8장에서 다루며, 이 결정에 필요한 판단 근거는 진단과 후속 사업의 규모 차이에서 얻을 수 있습니다.

사전 진단과 후속 사업의 규모

사전 진단은 3주 안팎으로 수행하는 작업이고, 그 뒤에 오는 상세설계와 본사업은 기간과 예산의 단위가 다릅니다. 2026년 상세설계 사업(2차)은 3개 기관의 3개 시스템을 대상으로 부가세 포함 약 13억 3천만 원, 계약일로부터 200일 규모이므로[S36] 시스템 하나에 평균 약 4억 4천만 원이 배정된 규모입니다. 2024년의 상세설계 사업은 4개 권역 합계 약 56억 원에 21개 시스템이어서 시스템당 평균 약 2억 7천만 원이었고[S35], NIA 가이드의 상세설계 수립비 산정 예시도 부가세 포함 2억 8,900만 원입니다[S45]. 즉 3주 규모의 진단 결과가 시스템당 수억 원이 드는 상세설계와 수십억 원 규모 본사업의 범위를 정하는 근거로 쓰입니다.

단계	기간	공개된 규모	이 단계에서 확정하는 항목
사전 진단	3주 안팎	해당 없음	가상화 예외 목록, 컨테이너 전환 공수 범위
상세설계	200일	3개 시스템 약 13억 3천만 원	목표모델, 이행계획, 소요 예산, 후속 사업 제안요청서
본사업	600일	3개 시스템 약 46억 원	전환 수행 8개월과 성능 유지관리 12개월의 이행

상세설계와 본사업의 수치는 각각 2026년 상세설계 2차 제안요청서와 2025년 지방교육청 본사업 제안요청서의 값입니다[S36][S41].

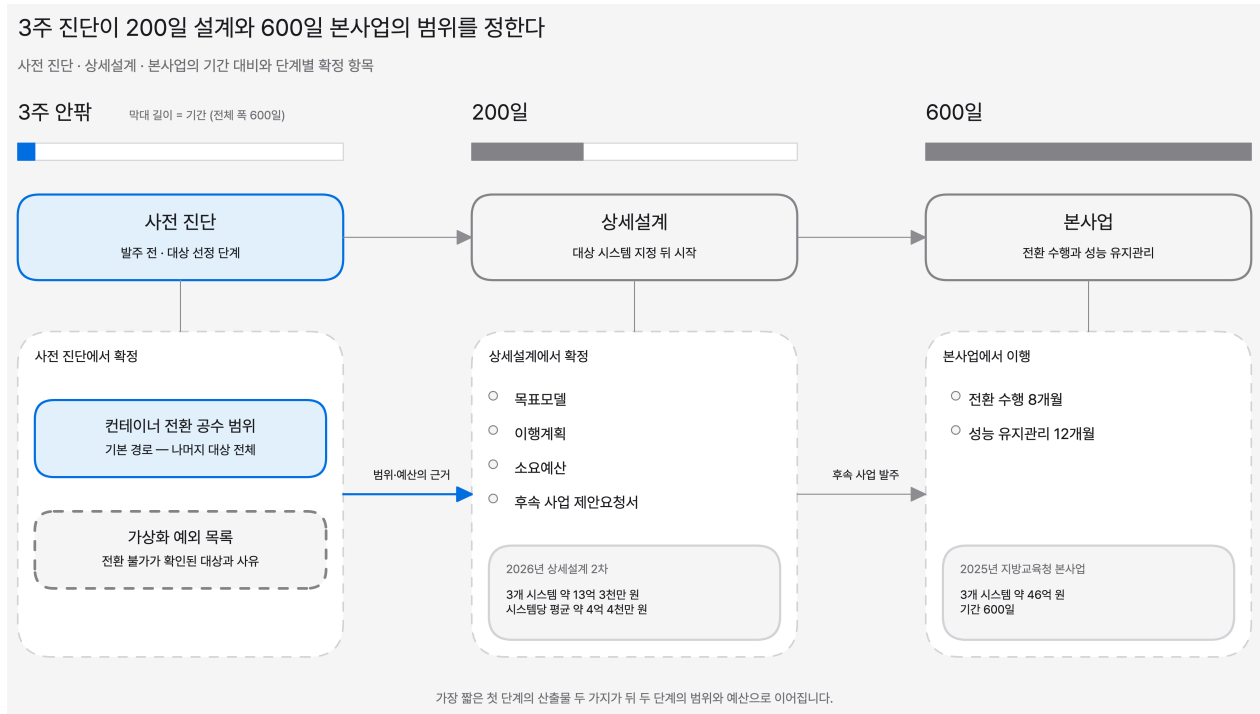


그림. 사전 진단, 상세설계, 본사업의 기간과 단계별 확정 항목

상세설계 앞 단계에 없는 기준

정부의 상세설계 사업은 대상 시스템이 지정된 뒤에 시작합니다. 과업은 현황분석, 목표모델 수립, 이행계획 수립, 예산 산정과 후속 사업 제안요청서 도출의 네 단계로 구성되고, 2026년 제안요청서에는 소스코드 검토를 포함해 누락 없이 조사하라는 요구가 새로 들어갔습니다[S36]. 그러나 어느 시스템을 상세설계 대상으로 올릴지, 그 시스템을 컨테이너로 전환할 수 있는지, 난이도가 어느 정도인지를 판단하는 기준은 이 문서에 없습니다. 이 판단은 상세설계보다 앞 단계에서 내려야 하고, CNTA는 그 판단에 필요한 수치를 제공합니다.

수치로 제시되는 진단 결과

CNTA는 판단 근거를 수치로 제시합니다. 정적 분석 도구가 소스코드와 배포 파일에서 수정이 필요한 부분을 찾아 이슈마다 Story Point를 매기는데, Story Point는 수정 난이도를 나타내는 점수입니다. 진단은 이 점수를 작업 일수로 환산해 시스템별 컨테이너 전환 공수를 산정하고, 컨테이너로 전환할 수 없는 대상은 사유와 함께 예외 목록으로 분리합니다. 의사결정권자는 이 결과로 어느 시스템을 먼저 상세설계에 올릴지를 비교할 수 있으며, 점수 체계와 환산 방법은 3장과 4장에서 설명합니다.

1.3 2030년까지 이어지는 공공 전환 일정과 순차 전환의 근거

공공부문의 클라우드 전환은 2030년을 목표 연도로 둔 다년 일정으로 추진되고 있습니다. 2023년 9월에 공개된 공공서비스 혁신 컨설팅 발표 자료는 전환율 목표를 2024년 10%, 2026년 45%, 2030년 100%로 제시했고[S46], 클라우드 네이티브 전환 상세설계 사업은 2024년에 네 차례 발주된 데 이어 2026년에도 공고가 나왔습니다[S35][S36]. 따라서 대상 기관은 여러 해에 걸쳐 여러 시스템을 전환해야 하며, 사전 진단은 개별 사업의 선택 사항이라기보다 다년 계획의 첫 단계에 해당합니다.

1.3.1 2026년 이후 신규 전환 물량에 대한 클라우드 네이티브 우선 적용 방침

연도별 전환율 목표

공공부문 전환 일정의 기준 수치는 전환율 목표입니다. 2023년 9월의 공공서비스 혁신 컨설팅 발표 자료에 실린 목표는 2024년 10%, 2026년 45%, 2030년 100%입니다[S46]. 이 목표대로라면 2026년의 45%에서 2030년의 100%까지 남은 55%포인트, 즉 전체의 절반이 넘는 물량을 2026년 이후에 전환해야 합니다. 2026년 이후에 전환 계획을 세우는 기관은 가장 많은 물량이 물리는 시기에 사업자와 예산을 확보해야 합니다.

발주 문서로 확인되는 우선 적용

클라우드 네이티브를 우선 적용한다는 방침은 실제 발주 문서의 목표 수준에서 확인할 수 있습니다. 2026년 상세설계 제안요청서와 2025년 지방교육청 본사업 제안요청서는 모두 성숙도 Level 3, 즉 클라우드 네이티브 단계를 목표로 적고 있습니다[S36][S41]. 2026년 상세설계 대상은 Unix와 구형 상용 WAS·DBMS로 구성된 시스템이어서, 리눅스와 x86 서버 위주였던 2024년 1차 사업보다 전환이 어려운 대상으로 범위가 넓어졌습니다[S36][S37]. 다만 신규 전환 물량 가운데 클라우드 네이티브를 적용하는 비율은 인용하는 자료마다 다르게 적혀 있으므로, 여기에는 발표 시점과 출처가 확인되는 전환율 목표만 실었습니다.

2021년부터 2030년까지의 주요 일정

정책과 발주의 흐름을 연도순으로 정리하면 준비, 설계, 전환의 순서가 드러납니다. 2021년에는 NIA가 발주자 안내서와 교육자료를 발간했고[S43][S44], 같은 해 공공부문 클라우드 도입 컨설팅에서는 심층 컨설팅 대상이 36개 기관 38건이었습니다[S31]. 2024년에는 정부가 상세설계 사업 네 건을 발주했고, 2025년에는 구축·운영 가이드 발간과 본사업 발주가 이어졌으며, 2026년에는 상세설계 2차 사업 공고가 나왔습니다[S35][S45][S41][S36].

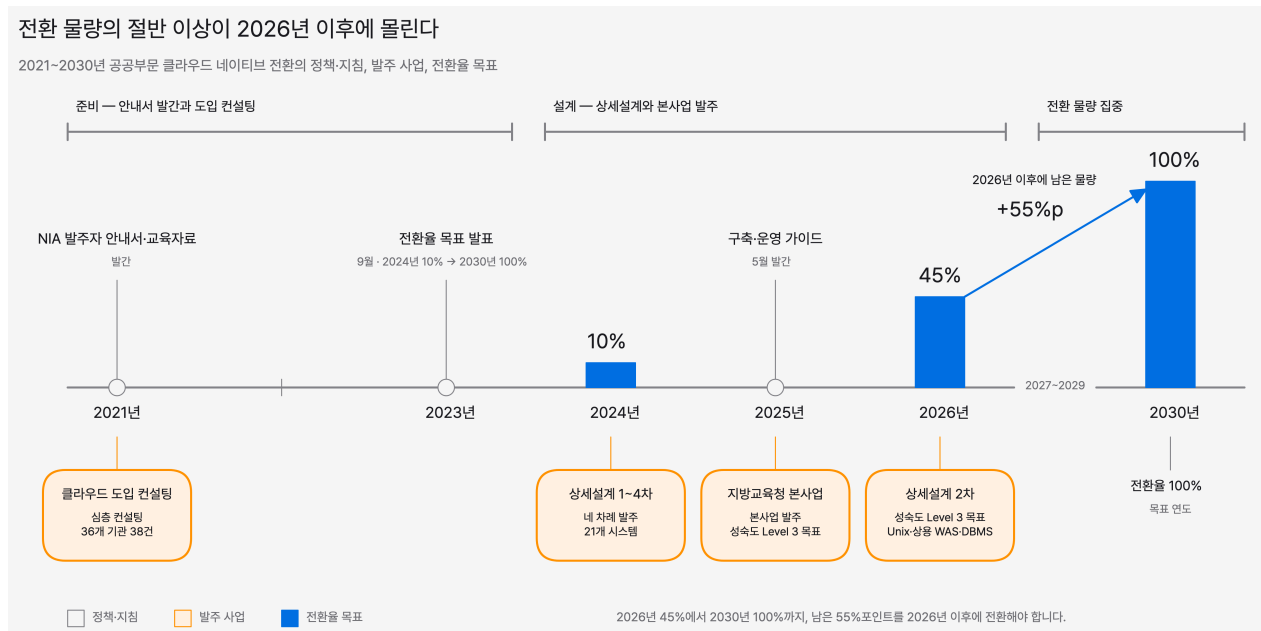


그림. 2021년부터 2030년까지 공공부문 클라우드 네이티브 전환의 주요 일정

대상 기관이 정할 전환 순서와 방식

이 일정에서 대상 기관이 정해야 하는 것은 전환 순서와 전환 방식입니다. 전환 여부는 2030년 100%라는 목표로 이미 정해져 있으므로, 남은 결정은 어느 시스템을 어느 해에 어떤 방식으로 전환할지입니다. 순서를 정하러

면 시스템별 난이도를 비교할 수 있어야 하고, 방식을 정하려면 컨테이너로 전환할 수 있는 대상과 예외 대상을 구분해야 합니다. 두 가지 모두 사전 진단의 산출물에서 나옵니다.

1.3.2 일괄 전환의 위험과 진단 후 순차 전환 방식

일괄 전환의 위험 요소

기관이 보유한 전 시스템을 한 사업에서 한 번에 전환하면 난이도가 다른 시스템이 같은 일정과 같은 예산 기준에 묶입니다. 2026년 상세설계 대상을 보면 3개 시스템 가운데 2개가 Unix를 주력으로 사용하고, 3개 모두 상용 WAS와 상용 DBMS를 쓰며, JDK 버전은 세 시스템 모두 문서에 없습니다[S36]. 이런 대상을 난이도 구분 없이 함께 전환하면 가장 어려운 시스템의 지연이 전체 개통 일정을 늦춥니다. AWS도 대규모 이전 가이드에서 구조를 고치는 Refactor 방식은 대규모 이전에 권장하지 않는다고 밝히고 있어[S33], 많은 시스템을 한꺼번에 재구성하는 계획은 해외 사업자의 권고와도 어긋납니다.

난이도 구분에 따른 순차 전환

순차 전환은 진단으로 시스템의 난이도를 나눈 뒤 쉬운 대상부터 순서대로 전환하는 방식입니다. NIA의 2025년 가이드도 클라우드 네이티브 도입 단계를 비핵심 시스템부터 적용하는 단계로 설명하고[S45], 가이드의 체크리스트가 제시하는 도입 필요단계는 최종 결과가 아니며 상세설계에서 정해진다고 덧붙입니다[S45]. 즉 정부 가이드는 순서를 두고 전환하는 방식을 전제로 하지만, 순서를 정하는 정량 기준은 제공하지 않습니다. 아래 표는 두 방식에서 생기는 위험 요소를 비교한 것입니다.

위험 요소	일괄 전환	진단 후 순차 전환
일정	가장 어려운 시스템이 전체 개통 시점을 결정	난이도가 낮은 대상부터 개통하고 뒤 단계 일정을 조정
예산	난이도 구분 없이 한 번에 편성해 오차가 누적	앞 단계의 실제 공수로 뒤 단계 예산을 보정
기술 검증	전환 방식의 문제가 전 시스템에서 동시에 발생	첫 대상에서 확인한 문제를 다음 대상에 반영
가상화 예외	사업 수행 중에 발견되어 구성 변경으로 처리	사전에 목록으로 분리해 별도 계획으로 관리

진단 결과의 세 가지 용도

순차 전환을 선택하려면 순서를 정할 근거가 있어야 하고, 사전 진단의 결과가 그 근거입니다. 진단 결과는 총사업비 산정, 위험도 평가, 단계별 추진 계획이라는 세 가지 작업의 입력값으로 쓰입니다. 예를 들어 시스템별 전환 공수는 총사업비의 애플리케이션 수정비 항목을 채우고, 예외 목록과 난이도 분포는 위험도 평가와 단계 구성의 기준이 됩니다. 진단 결과로 우선순위를 정하고 단계를 구성하는 방법은 8장에서 이어서 다룹니다. 그에 앞서 2장은 사전 진단이 답하는 질문과 절차, 산출물을 설명합니다.

2장. 사전 진단 CNTA의 정의와 절차: 다섯 단계와 단계별 산출물

1장에서 본사업 발주 전에 확정할 두 가지로 든 가상화 예외 목록과 컨테이너 전환 공수는 사전 진단 CNTA의 산출물입니다. CNTA는 대상 애플리케이션을 컨테이너로 옮길 수 있는지, 옮기려면 기간·인력·공수가 얼마나 드는지를 수치로 답합니다. 정부도 전환 본사업 앞에 상세설계 사업을 따로 발주하지만, 2026년 2차 상세설계 제안 요청서는 전환유형을 나누는 기준을 「전환시 난이도, 수정의 범위」로만 적고 난이도 등급표와 대상 선정 기준은 사업자의 과업으로 둡니다[S36]. 따라서 어느 시스템을 어떤 방식으로 옮길지와 그 공수가 얼마인지는 발주 기관이 사업 전에 따로 확인해 두어야 합니다. 그 확인은 사전 진단이 답하는 질문 여섯 가지, 다섯 단계 절차와 단계별 산출물, 3주 일정의 구성 예, 기관이 준비할 항목으로 구체화됩니다.

2.1 CNTA(Cloud Native Transformation Assessment)가 답하는 여섯 가지 질문

CNTA는 진단 범위를 질문 여섯 가지로 고정합니다. 여섯 가지는 전환 가능한 업무, 난이도, 소요 기간, 재개발 필요 여부, 목표 환경과 WAS, 개발 조직에 전달할 요구사항입니다. 질문 목록이 계약 전에 정해져 있으면 발주 기관은 결과 보고서에서 받아야 할 항목을 검수 기준으로 미리 적어 둘 수 있습니다. 반면 범위를 질문으로 정하지 않은 진단은 조사 항목이 수행자마다 달라지므로 결과를 다음 사업의 예산 근거로 쓰기 어렵습니다.

2.1.1 전환 가능 여부·난이도·기간을 수치로 답하는 사전 진단의 범위

애플리케이션별 수치가 사전 진단의 핵심 범위

사전 진단은 어떤 업무를 옮길 수 있는지, 난이도가 어느 정도인지, 얼마나 걸리는지를 애플리케이션 하나하나에 대해 수치로 제시합니다. 난이도의 단위인 Story Point는 수정 작업의 상대적인 크기를 나타내는 점수이며, 애플리케이션 현대화 분석 도구는 소스나 배포 파일에서 찾은 이슈 한 건마다 0·1·3·5·7·13점 가운데 하나를 부여합니다. 애플리케이션별 총점은 이슈의 발견 횟수에 건당 점수를 곱해 계산합니다. 도구가 낸 점수에 목표 환경과 업무 크기를 반영해 일수를 산정하므로, 사전 진단은 난이도에 대한 의견 대신 애플리케이션별 점수와 일수를 결과로 냅니다.

수치 없는 진단 결과는 총사업비의 근거가 되지 못한다

진단 결과는 총사업비 산정, 위험도 평가, 단계별 추진 계획의 입력값으로 쓰이므로 정성 의견만 담긴 결과로는 시스템별 예산을 나눌 수 없고 전환 순서도 정하기 어렵습니다. NIA의 2025년 클라우드 네이티브 구축·운영 가이드도 적합성 체크리스트가 제시하는 도입 필요 단계를 최종 결과로 보지 않고, 상세설계를 거쳐야 정해지는 값으로 설명합니다[S45]. 2026년 2차 상세설계 제안요청서도 현황분석 요구사항에 소스코드 검토를 포함한 전수 조사를 넣었습니다[S36]. 즉 발주 문서도 코드 수준의 조사 결과를 예산의 출발 자료로 요구하고 있습니다.

질문 여섯 가지와 산출물의 대응

질문마다 답의 형태와 그 답이 실리는 산출물이 정해져 있으므로, 아래 표를 계약서의 과업 범위에 그대로 옮기면 결과 보고서의 검수 항목으로 쓸 수 있습니다.

질문	답의 형태	답이 실리는 산출물
어떤 업무를 옮길 수 있는가	애플리케이션별 컨테이너 전환 가능 여부, 예외 사유	애플리케이션 인벤토리, 전환 전략 표

질문	답의 형태	답이 실리는 산출물
난이도는 어느 정도인가	Story Point 총점, 이슈 분류별 건수	자동화 분석 보고서
얼마나 걸리는가	애플리케이션별 일수	진단 결과 보고서
재개발이 필요한가	재개발 대상 목록과 근거	전환 전략 표
목표 환경과 WAS는 무엇인가	목표 Java·WAS·OS 버전	목표 구성안
개발 조직에 무엇을 요구하는가	수정 항목 목록	진단 결과 보고서

여섯 질문 가운데 전환 가능 여부는 컨테이너 전환을 기본으로 묻습니다. 사전 진단은 대상마다 컨테이너로 옮길 수 있는지를 먼저 판정하고, 옮길 수 없는 사유가 확인된 대상만 가상화 전환 예외 목록에 포함합니다. 진단이 끝나면 발주 기관은 컨테이너 전환 대상의 공수와 예외 목록을 함께 받으며, 예외를 허용하는 사유의 기준은 6장에서 다룹니다.

2.1.2 재개발 필요 여부와 목표 환경·WAS 선정, 개발 조직 요구사항

목표 Java·WAS 버전에 따라 달라지는 전환 공수

목표 환경을 정하지 않은 공수 산정은 성립하지 않는데, 같은 애플리케이션이라도 목표 버전에 따라 수정해야 하는 코드의 범위가 달라지기 때문입니다. Jakarta EE 9는 2020년 12월 8일 릴리스에서 패키지 네임스페이스(클래스 이름 앞에 붙는 패키지 경로)를 javax에서 jakarta로 바꿨고[S12], Apache Tomcat은 10 버전부터 jakarta 네임스페이스를 쓰며 Tomcat 9가 Java EE 8을 지원하는 마지막 메이저 버전입니다[S15]. 예를 들어 목표 WAS를 Tomcat 9로 정하면 네임스페이스 변경 작업이 범위에서 빠지고, Apache Tomcat 10.1이나 오픈소스 Jakarta EE 서버로 정하면 그 작업이 전부 범위에 들어갑니다. Spring Boot 3.0도 Java 17과 Spring Framework 6을 요구하므로 프레임워크 목표 버전이 Java 목표 버전을 함께 정합니다[S13].

재개발 대상의 판별 기준이 되는 Story Point 등급

재개발이 필요한 애플리케이션은 분석 도구가 이슈마다 부여한 등급으로 판별합니다. 분석 도구의 등급표에서 7점은 컴포넌트나 서브시스템을 다시 설계해야 하는 경우이고, 13점은 해결책이 알려지지 않아 전체를 다시 작성할 수 있는 경우입니다. 정부 상세설계 제안요청서는 Refactor(재구성) 유형을 일부 개정과 전면 재개발로 나누고, 사업 기간 안에 끝낼 수 없는 전면 재개발은 대상에서 제외합니다[S36]. 높은 등급의 이슈가 어느 애플리케이션에 집중돼 있는지를 본사업 전에 확인해야 사업 범위에 넣을 대상과 뺄 대상을 정할 수 있습니다.

개발 조직에 전달하는 코드 수정 요구사항

분석 도구는 고치지 않으면 빌드나 실행이 되지 않는 항목을 Mandatory로 분류하며, 대표 예는 목표 환경이 지원하지 않는 특정 제품 전용 인터페이스를 호출하는 코드입니다. 사전 진단은 이 항목을 하드코딩된 IP와 파일 경로, 로컬 파일 입출력, 특정 OS에 의존하는 코드와 함께 줄 단위로 정리해 개발 조직의 수정 요구사항으로 전달합니다. 요구사항에는 수정할 파일과 위치가 함께 적히므로 개발 조직은 이 목록으로 수정 작업량을 미리 가늠하고 담당자를 배정할 수 있습니다.

진단 전에 기관 내부에서 합의할 목표 환경 결정 항목

IT 담당자는 진단을 시작하기 전에 아래 표의 결정 항목을 기관 내부에서 합의해 두는 것이 좋습니다. 합의가 없으면 진단 수행자가 목표 버전을 가정해 계산하게 되고, 가정이 바뀔 때마다 공수를 다시 산정해야 합니다.

결정 항목	선택 예	공수에 미치는 영향
목표 Java 버전	OpenJDK 11·17·21	제거된 모듈, 내부 접근 제한 항목의 포함 여부
목표 WAS	Apache Tomcat 10.1, 오픈소스 Jakarta EE 서버	jakarta 네임스페이스 변경의 포함 여부
목표 프레임워크	Spring Boot 3.x	Java 17 이상 전제, 라이브러리 호환 검증
목표 OS	리눅스 배포판과 버전	컨테이너 이미지와 호스트의 버전 조합

목표 Java 버전은 메이저 버전에 더해 업데이트 번호까지 정해야 합니다. JVM(Java Virtual Machine, Java 가상 머신)이 컨테이너의 자원 한도를 기본으로 인식하는 것은 JDK 10부터이고 JDK 8 계열에서는 8u191부터이며[S07], 버전별 영향은 5장에서 다룹니다.

2.2 사전 조치에서 결과 공유까지 다섯 단계의 활동과 산출물

CNTA는 사전 조치, 현황 분석, 목표 모델링, 전환 전략 수립, 결과 공유의 다섯 단계로 진행하며 단계마다 기관이 받는 산출물이 정해져 있습니다. 첫 단계인 사전 조치에서는 고객 기관이 현황 분석 체크리스트를 작성하며, 행정·공공기관 클라우드 네이티브 전환지원 사업의 계약특수조건도 전환대상기관에 OS, WEB/WAS, DBMS 전환 사항을 담은 전환 체크리스트의 최종 확인 의무를 둡니다[S42]. 체크리스트의 내용 확인은 본사업에서도 기관이 지는 의무이므로 사전 진단에서 미리 작성해 두면 본사업의 준비 자료로 쓸 수 있습니다.

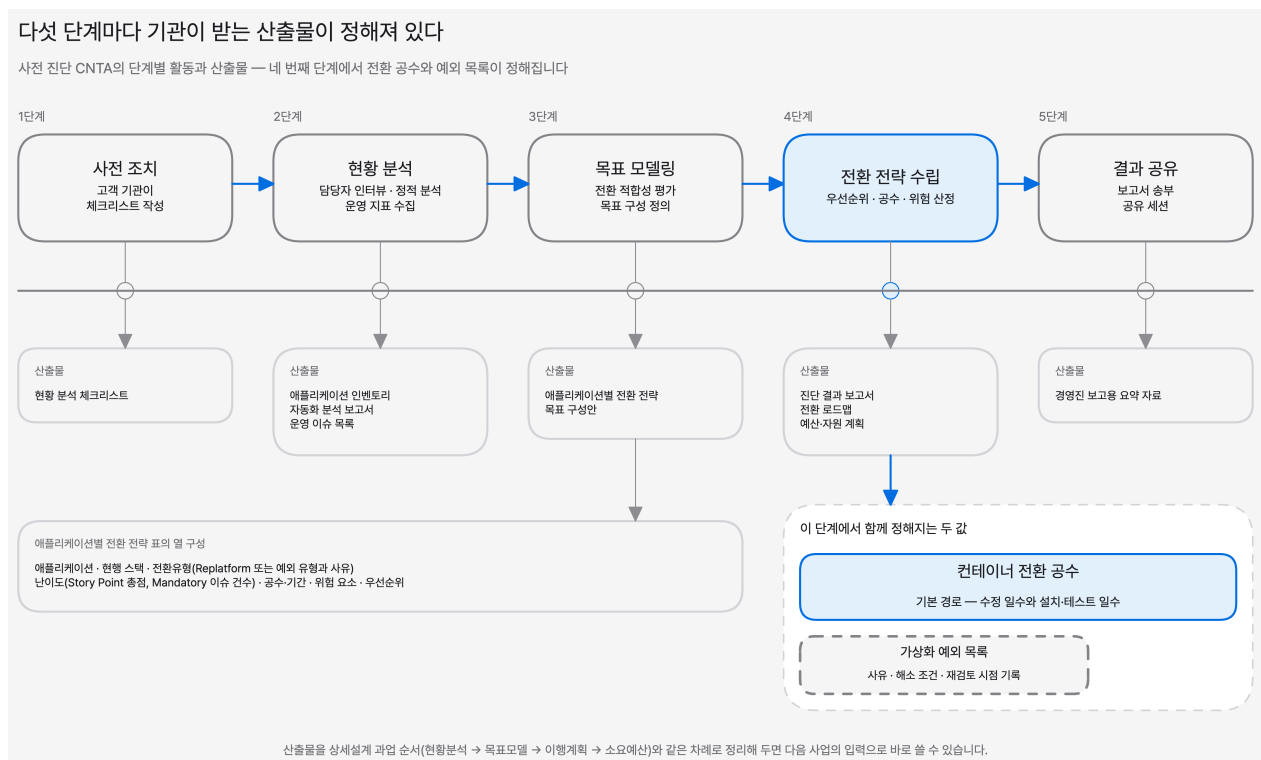


그림. 사전 진단 CNTA의 다섯 단계와 단계별 활동·산출물

다섯 단계는 정부 상세설계 사업의 과업 순서와 대응합니다. 2026년 2차 상세설계 제안요청서의 과업은 현황 분석, 목표모델 수립, 전환 이행계획, 소요예산 산출과 후속 사업 제안요청서 작성의 순서를 따릅니다[S36]. 사전 진단의 산출물을 같은 순서로 정리해 두면 상세설계 사업자가 그 자료를 과업별 입력으로 바로 쓸 수 있습니다.

2.2.1 현황 분석 단계의 담당자 인터뷰·정적 분석·운영 지표 수집

체크리스트를 바탕으로 한 담당자 인터뷰

현황 분석은 고객 기관이 작성한 체크리스트를 진단 수행자가 담당자와 함께 확인하는 인터뷰로 시작합니다. 인터뷰에서는 세션 처리 방식, 상용 소프트웨어 제작사의 컨테이너 지원 여부, 하드웨어 의존, 적용받는 규제처럼 코드만 보아서는 알 수 없는 사항을 확인합니다. 진단 수행자는 이 결과를 애플리케이션 인벤토리(애플리케이션 목록과 구성 정보)로 정리하고 애플리케이션별 OS, WAS, Java 버전, 연계 시스템을 적으며, 체크리스트의 구성은 3장에서 다룹니다.

소스 또는 배포 파일의 정적 분석

정적 분석은 프로그램을 실행하지 않고 소스 코드나 배포 파일을 규칙에 따라 검사하는 방법입니다. 분석 대상은 소스 또는 EAR·WAR·JAR 형식의 Java 배포 파일이며, 분석 도구는 애플리케이션별 총점, 이슈 분류별 집계, 사용 기술 목록, 의존성, 하드코딩된 IP 목록을 보고서로 냅니다. 2026년 2차 상세설계 제안요청서도 현황조사를 소스코드 검토 등으로 누락 없이 수행하라고 요구합니다[S36]. 따라서 사전 진단에서 만든 자동화 분석 보고서는 상세설계의 현황분석 요구사항에 그대로 대응합니다.

운영 지표 수집과 운영 이슈 목록

운영 지표는 동시 사용자 수, 평균 응답시간, 현행 운영 이슈처럼 현재 운영 상태를 나타내는 자료입니다. 2025년에 발주된 지방교육청 통합누리집 전환 사업의 제안요청서는 전환 뒤에도 기존과 동등한 수준의 성능을 보장 하라고 요구하고, 성과 비교 기준을 개통 전 3년간 데이터의 평균으로 둡니다[S41]. 전환 전의 운영 지표를 진단 단계에서 수집해 두어야 본사업에서 이 성능 기준을 증명할 수 있으며, 진단 수행자는 수집 결과를 운영 이슈 목록으로 정리합니다.

현황 분석의 세 활동이 확인하는 내용과 그 결과가 실리는 산출물은 다음 표와 같이 대응합니다.

활동	확인하는 것	산출물
담당자 인터뷰	구성, 연계, 세션 방식, 상용 소프트웨어 지원 여부	애플리케이션 인벤토리
정적 분석	이슈 목록, Story Point, 사용 기술, 의존성	자동화 분석 보고서
운영 지표 수집	동시 사용자 수, 평균 응답시간, 현행 운영 이슈	운영 이슈 목록

2.2.2 목표 모델링과 전환 전략 수립 단계의 애플리케이션별 전환 전략

컨테이너 전환을 기본으로 하는 전환 적합성 평가

목표 모델링 단계에서는 현황 분석 결과로 전환 적합성을 평가하고 목표 아키텍처와 기술 스택을 정하며, 평가는 대상마다 컨테이너로 옮길 수 있는지를 먼저 판정하는 순서로 진행합니다. 컨테이너로 옮길 수 없는 사유가 확인된 대상만 가상화 전환 예외로 분류하고, 예외마다 사유와 해소 조건, 재검토 시점을 기록합니다. NIA의 2025년 가이드는 전환유형을 고르는 정량 기준을 제시하지 않고 사업자가 기준과 절차를 수립하도록 하므로[S45], 이 판정 기준을 사전 진단에서 먼저 마련해 두면 상세설계의 출발 자료로 쓸 수 있습니다.

정부 제안요청서의 정의에 맞춘 전환유형 용어

전환유형 용어는 공공기관 독자가 발주 문서와 바로 대조할 수 있도록 정부 제안요청서의 정의에 맞춥니다. 제안요청서는 시스템 소프트웨어 버전 변경만으로 옮기는 유형을 Rehost(단순전환)로, OS·WEB·WAS·DBMS 같은 플랫폼 변경으로 소스 수정과 데이터 변환이 따르는 유형을 Replatform(플랫폼전환)으로 정의합니다. 변경 정도에 따라 일부 개정이나 전면 재개발이 필요한 유형은 Refactor(재구성)로 분류합니다[S36]. AWS와 Microsoft가 공개한 전환 전략 분류에서도 컨테이너화는 Replatform에 속합니다[S33][S34]. 정부 제안요청서의 정의로 보아도 컨테이너 전환은 플랫폼을 바꾸면서 소스를 수정하는 Replatform에 가장 가깝습니다. 따라서 이 백서는 컨테이너 전환을 Replatform으로 적으며, 유형별 판정 기준은 6장에서 다룹니다.

애플리케이션별 공수·기간·위험 요소와 우선순위 산정

전환 전략 수립 단계에서는 애플리케이션을 묶어 우선순위를 정하고 공수, 기간, 비용, 위험 요소를 산정하며, 이 단계에서 컨테이너 전환 공수와 가상화 전환 예외 목록이 함께 정해집니다. 다만 Story Point는 분석 도구의 공식 문서가 밝히듯 인시(man-hour)로 바로 환산되는 값이 아니므로, 일수는 목표 버전과 업무 크기를 반영해 따로 산정하며 그 방법은 4장에서 다룹니다.

애플리케이션별 전환 전략 표는 판정 결과와 그 근거를 한 줄에서 확인할 수 있도록 아래 열로 구성합니다.

열	적는 내용
애플리케이션	업무명, 유형(웹·배치·연계)
현행 스택	OS, WAS, Java, 프레임워크 버전
전환유형	Replatform(컨테이너 전환) 또는 예외 유형과 사유
난이도	Story Point 총점, Mandatory 이슈 건수
공수·기간	수정 일수, 설치·테스트 일수
위험 요소	분석하지 못한 모듈, 상용 소프트웨어 의존, 연계
우선순위	전환 순서와 묶음

2.2.3 진단 결과 보고서·전환 로드맵·예산 계획으로 구성되는 최종 산출물

최종 산출물 세 가지와 경영진 보고용 요약 자료

CNTA의 최종 산출물은 진단 결과 보고서, 전환 로드맵, 예산·자원 계획이며, 결과 공유 단계에서는 보고서를 송부하고 공유 세션을 열어 경영진 보고용 요약 자료를 함께 제공합니다. 진단 결과 보고서는 애플리케이션별 전환 전략 표와 자동화 분석 보고서를 포함하고, 전환 로드맵은 전환 순서와 단계 구분을 제시합니다. 예산·자원 계획은 공수와 기간, 필요한 인력 구성을 담아 다음 사업의 소요예산 산출에 쓰입니다.

후속 사업의 입력 자료로 쓰이는 산출물 형태

산출물은 상세설계와 본사업 제안요청서의 입력 자료로 쓸 수 있는 형태여야 하며, 형태가 맞지 않으면 다음 사업에서 같은 조사를 다시 하게 되고 그 비용은 발주 기관이 부담합니다. 2024년 3차 상세설계 사업은 2022년에 받은 심층 컨설팅 결과를 환경 변화를 고려해 재검토하는 과업을 넣었고[S39], 4차 사업은 기존 컨설팅을 받은 4개 기관의 소요예산을 검증하고 다시 산출하는 과업을 넣었습니다[S40]. 즉 앞 단계의 진단 결과를 그대로 쓰지 못하고 다시 산정한 사례가 발주 문서에 남아 있습니다.

최종 산출물별 주요 내용과 후속 사업에서의 쓰임은 다음 표와 같으며, 산출물 이름은 검수 항목으로 그대로 쓸 수 있습니다.

산출물	주요 내용	후속 사업에서의 쓰임
진단 결과 보고서	인벤토리, 분석 보고서, 전환 전략 표, 예외 목록	상세설계 현황분석의 입력, 대상 선정 근거
전환 로드맵	전환 순서, 단계 구분	이행계획과 단계별 발주 범위
예산·자원 계획	공수, 기간, 인력 구성	소요예산 산출과 제안요청서 작성
경영진 보고용 요약 자료	판정 결과와 의사결정 항목	사업 추진 여부 결정

재산정 비용을 줄이는 산출물 검수 조건

발주 기관은 산출물을 검수할 때 수치와 함께 판정의 근거가 실렸는지를 확인해야 합니다. 목표 버전, 범위에서 제외한 항목, 예외 사유가 보고서에 적혀 있으면 다음 사업자가 같은 전제에서 계산을 이어 갈 수 있습니다. 반면 결론 수치만 적힌 보고서는 전제가 바뀔 때 검증할 방법이 없으므로 처음부터 다시 산정해야 합니다.

2.3 3주 진단 일정의 구성과 투입 역할, 고객 기관의 사전 준비물

CNTA는 시스템 한 묶음을 대상으로 할 때 3주로 구성할 수 있으며, 투입 역할은 다섯 가지이고 기관이 준비할 항목은 여섯 가지입니다. 정부 상세설계 사업은 대상 시스템이 지정된 뒤에 시작하며, 2026년 2차 사업은 3개 시스템을 대상으로 하고 사업 기간이 계약일로부터 200일입니다[S36]. 반면 사전 진단은 그 앞 단계에서 대상 선정, 전환유형 분류, 공수 범위를 먼저 제시하는 단기 절차입니다. 발주 기관은 진단 기간과 기관 내부의 준비 부담을 함께 고려해 착수 시기를 정할 수 있습니다.

2.3.1 주차별 활동으로 본 3주 진단 일정의 구성 예

시스템 한 묶음을 대상으로 한 3주 일정

공공기관 한 곳의 시스템 한 묶음을 대상으로 세운 사전 진단 수행 계획은 3주 일정이었습니다. 1주차에는 현황을 파악하고 1차 진단을 수행하며, 2주차에는 서비스 분리 타당성을 검토하고 시범 전환 후보를 선정하고, 3주차에는 결과를 통합해 로드맵 초안을 작성합니다. 3주는 하나의 구성 예이므로 실제 기간은 대상 시스템의 수와 규모에 맞춰 조정합니다.

주차	활동
1주차	현황 이해와 1차 진단

주차	활동
2주차	서비스 분리 타당성 검토, 시범 전환 후보 선정
3주차	결과 통합, 로드맵 초안 작성

사전 진단에 투입하는 다섯 가지 역할

진단에는 사업 관리자, 클라우드 네이티브 아키텍트, 도메인 분석가, 분석 도구 전문가, 공공 정책 자문이 참여합니다. 역할 분담을 예로 들면 사업 관리자는 일정과 기관 협의를 맡고, 아키텍트는 목표 구성과 전환유형 판정을 맡습니다. 도메인 분석가는 업무 단위의 서비스 분리 타당성을 검토하고, 분석 도구 전문가는 정적 분석을 수행해 결과를 해석하며, 공공 정책 자문은 발주 문서와 가이드의 요구사항을 대조합니다. 역할별 인원은 대상 시스템의 규모에 따라 달라지므로 착수 전에 기관과 협의해 정합니다.

대상 수와 규모에 따른 진단 기간의 증감

기간은 대상 시스템의 수와 규모가 커질수록 늘어나며, 보안 구역의 반입·반출 절차에 걸리는 시간도 기간에 더해집니다. 여러 시스템을 한 번에 진단하려는 기관은 시스템 묶음별로 일정을 나누어 순서대로 진행하는 구성을 검토할 수 있습니다.

2.3.2 분석용 PC·OpenJDK 11 이상·애플리케이션 반입으로 구성되는 진단 전 준비 항목

진단 착수 전에 기관이 준비하는 여섯 가지 항목

진단을 시작하려면 기관이 아래 여섯 가지를 준비해야 하며, IT 담당자는 이 점검표를 내부 요청서에 그대로 옮겨 쓸 수 있습니다.

준비 항목	준비 내용
일정 협의	진단 기간과 인터뷰 일정
분석용 PC	진단 전용으로 쓰는 별도 작업 PC
OpenJDK 11 이상	분석용 PC에 설치
대상 애플리케이션 반입	소스 또는 EAR·WAR·JAR 배포 파일
분석 도구 반입	보안 구역 반입 승인
결과 보고서 반출	분석 보고서 반출 권한과 절차

분석 도구 실행 환경과 대상 애플리케이션의 구분

OpenJDK 11 이상은 분석 도구를 실행하는 데 필요한 조건이며, 진단 대상 애플리케이션의 Java 버전과는 관계가 없습니다. Java 1.6이나 1.7로 작성된 애플리케이션도 분석용 PC에 OpenJDK 11 이상만 있으면 분석할 수 있습니다. 다만 반입하는 파일은 업무 애플리케이션 자체여야 하며, 대학 E의 진단에서는 반입된 파일 가운데 2건이 타사 라이브러리 JAR이거나 업무용이 아닌 WAR여서 분석 결과를 내지 못했습니다.

보안 구역 반입·반출 절차와 진단 일정의 관계

보안 구역의 반입·반출 절차는 진단 일정의 선행 조건이므로 승인 일정을 진단 일정보다 먼저 확인해야 합니다. 분석 도구와 대상 애플리케이션이 분석용 PC에 준비되지 않으면 정적 분석을 시작할 수 없고, 보고서 반출 승인이 늦어지면 결과 통합이 늦어집니다. 준비가 늦어진 기간만큼 진단 기간이 늘어나므로 반입·반출 승인은 착수일 이전에 끝내 두는 것이 안전합니다.

2.4 클라우드 사업자의 준비도 평가와 코드 기반 사전 진단의 차이

해외 클라우드 사업자가 제공하는 이전 준비도 평가와 코드 기반 사전 진단은 묻는 질문이 다릅니다. 준비도 평가는 조직이 클라우드로 이전할 준비가 됐는지를 묻고, 사전 진단은 개별 애플리케이션의 수정 범위와 공수를 묻습니다. 준비도 평가를 이미 받은 기관이라도 애플리케이션별 전환 공수는 따로 확인해야 합니다. 국내 공공부문에서도 2021년 클라우드 도입 컨설팅에서 심층 컨설팅이 36개 기관 38건 진행된 바 있어[S31], 앞서 받은 컨설팅이 어느 질문까지 답했는지를 먼저 확인하는 것이 순서입니다.

2.4.1 조직 준비도를 묻는 워크숍형 평가의 범위

AWS·Microsoft·Google이 공개한 준비도 평가의 범위

AWS Migration Readiness Assessment는 4~6시간 워크숍에서 24개 영역 약 70문항으로 조직의 현재 위치와 강점·약점을 확인하고 격차 해소 계획을 냅니다[S28]. Microsoft Cloud Adoption Framework는 아키텍처, 구성요소, 의존성, 컴플라이언스, 코드, 데이터베이스, 위험 등록부의 순서로 워크로드 평가 절차를 제시합니다[S29]. Google Cloud의 Rapid Migration and Modernization Program은 데이터 기반 현황 조사와 TCO(Total Cost of Ownership, 총소유비용) 계산을 제공합니다[S30]. 세 평가 모두 해당 사업자의 클라우드로 이전하는 것을 전제로 합니다. 또한 투입 인력과 가격을 공개한 사업자는 없고, 공개된 기간도 AWS의 워크숍 시간뿐입니다.

평가 유형별 질문·기간·산출물 비교

평가 유형별로 묻는 질문, 공개된 기간, 산출물을 코드 기반 사전 진단과 나란히 놓으면 다음 표와 같습니다[S28][S29][S30].

평가 유형	묻는 질문	공개된 기간	산출물
AWS Migration Readiness Assessment	조직이 이전할 준비가 됐는가	4~6시간 워크숍	현재 위치, 강점·약점, 격차 해소 계획
Microsoft Cloud Adoption Framework 워크로드 평가	워크로드를 어떤 순서로 평가하는가	공개 수치 없음	평가 결과, 위험 등록부
Google Cloud Rapid Migration and Modernization Program	현황과 총소유비용은 얼마인가	공개 수치 없음	현황 조사 결과, TCO
코드 기반 사전 진단 (CNTA)	애플리케이션별 수정 범위와 공수는 얼마인가	3주 구성 예	진단 결과 보고서, 전환 로드맵, 예산·자원 계획

준비도 평가가 답하지 않는 애플리케이션별 수정 범위

준비도 평가는 조직이 클라우드 이전을 감당할 준비를 갖췄는지 확인하고 부족한 부분의 해소 계획을 세우는 데 쓰입니다. 하지만 개별 애플리케이션의 어느 코드를 고쳐야 하는지, 그 수정에 며칠이 드는지는 평가 범위에 들어 있지 않습니다. 두 평가는 서로 보완하는 관계이므로 준비도 평가로 조직 과제를 확인한 기관은 코드 기반 사전 진단으로 애플리케이션 과제를 이어서 확인할 수 있습니다.

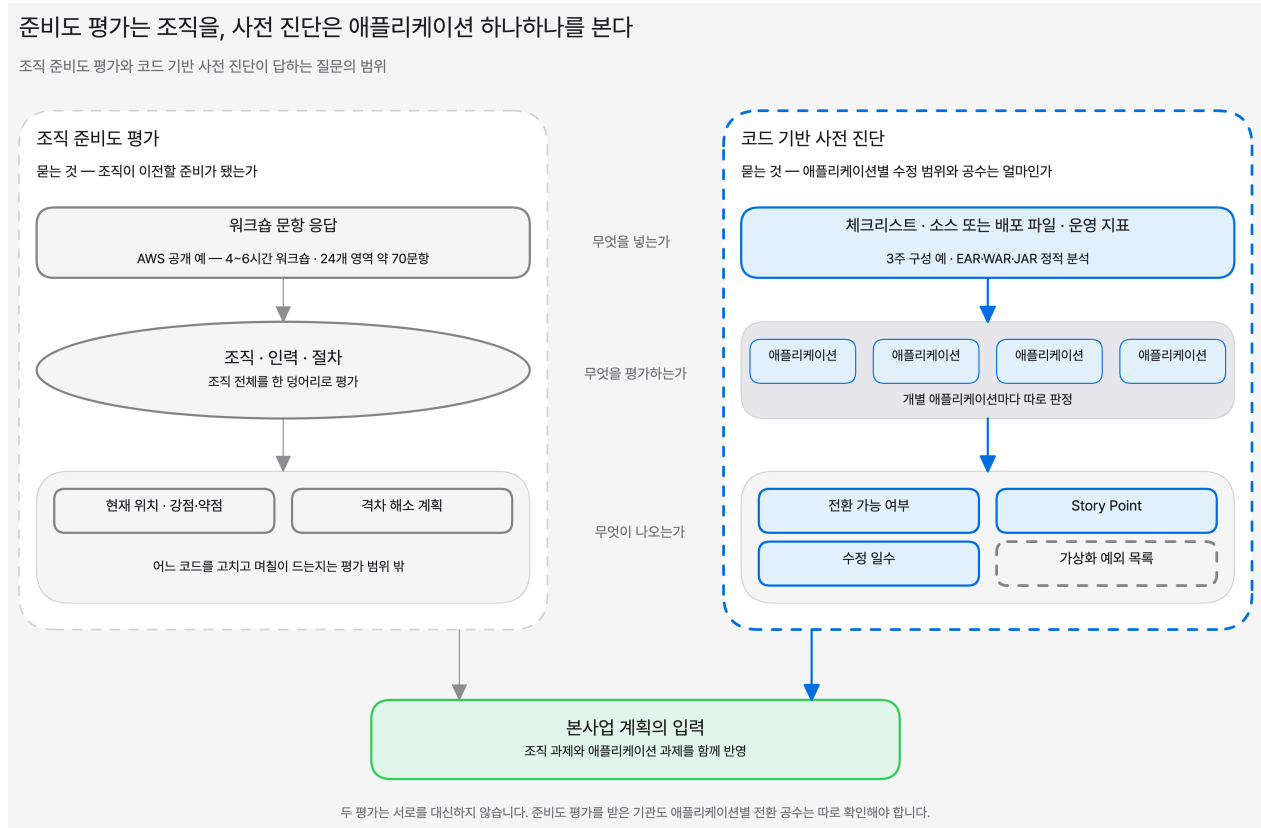


그림. 조직 준비도 평가와 코드 기반 사전 진단이 답하는 질문의 범위

2.4.2 분석 도구가 무료인 환경에서 해석과 공수 산정이 진단의 가치가 되는 이유

오픈소스로 공개된 Java 코드 분석 도구

Java 코드를 분석하는 애플리케이션 현대화 분석 도구는 오픈소스로 공개돼 있어 누구나 실행할 수 있습니다. Microsoft도 같은 오픈소스 엔진 계열의 Java 코드 평가 도구를 Azure Migrate에서 무료로 제공합니다 [S29]. 도구를 실행해 점수를 얻는 일은 누구나 할 수 있으므로 진단 영역의 가치는 결과 해석, 목표 버전 결정, 공수 산정, 예외 판정에서 나옵니다.

분석 도구의 점수만으로는 전환 건적이 나오지 않는다

분석 도구의 공식 문서는 Story Point가 반드시 인시로 환산되지는 않으며 작업 사이에 값이 일관되어야 한다고 설명합니다. 점수는 작업 사이의 상대 크기를 나타내는 값일 뿐이고 그 자체가 일수를 뜻하지는 않습니다. 예를 들어 공공기관 D의 애플리케이션은 총점 937점 가운데 920점이 목표 버전의 범위에서 제외할 수 있는 항목이 있었습니다. 목표 버전을 어떻게 정하느냐에 따라 같은 보고서의 공수가 크게 달라지므로 점수를 일수로 옮기려면 범위 판정이 먼저 있어야 하며, 이 사례는 4장에서 자세히 다룹니다.

분석 도구의 출력과 사전 진단이 더하는 판단

분석 도구가 보고서로 내는 항목과 사전 진단이 그 위에 더하는 판단을 나누면 다음 표와 같습니다.

분석 도구가 내는 것	사전 진단이 더하는 것
이슈 목록과 Story Point 총점	목표 버전에 따른 포함·제외 범위 판정
Mandatory·Optional·Potential·Information 분류	Potential 항목의 필수 여부 판정
사용 기술, 의존성, 하드코딩된 IP 목록	인터뷰로 확인한 세션 방식, 상용 소프트웨어 지원, 하드웨어 의존 반영
줄 단위 소스 보고서	개발 조직 요구사항과 애플리케이션별 일수
해당 없음	가상화 전환 예외 판정과 사유·재검토 시점 기록

분석 도구는 Potential 항목을 필수 여부를 판단할 정보가 부족한 항목으로 분류하므로, 이 항목은 진단 수행자가 직접 검토해 처리해야 합니다. 가상화 전환 예외도 코드 분석만으로는 판정할 수 없고 인터뷰에서 확인한 사실이 함께 필요합니다. 따라서 발주 기관이 진단 영역에서 검수해야 할 것은 판정의 근거와 일수 산정의 전제입니다. 그 근거를 만드는 체크리스트와 정적 분석의 측정 항목은 3장에서 다룹니다.

3장. 진단이 재는 것: 체크리스트 13개 분류와 정적 분석

2장에서 본 현황 분석 단계에서 사전 진단은 성격이 다른 두 가지 자료를 수집합니다. 하나는 전환대상기관이 직접 작성하는 현황 체크리스트와 담당자 인터뷰이고, 다른 하나는 애플리케이션 현대화 분석 도구가 소스 또는 바이너리를 규칙으로 검사해서 내는 정적 분석 결과입니다. 체크리스트는 서버 구성이나 세션 처리 방식처럼 코드에 드러나지 않는 사실을 담고, 정적 분석은 수정이 필요한 코드의 위치와 건수를 줄 단위로 담습니다. 따라서 컨테이너 전환이 가능한지와 공수가 얼마나 드는지는 두 자료를 합쳐야 판정할 수 있습니다.

두 자료를 함께 수집해야 하는 이유는 공개된 가이드의 체크리스트가 기술 현황을 충분히 묻지 않는다는 데 있습니다. 2021년 12월에 나온 「클라우드 네이티브 정보시스템 구축 발주자 안내서」의 적합성 검토 체크리스트는 정책 변화 대응, 잦은 변경 요청, 장애 경험처럼 도입 필요성을 묻는 10문항으로 구성됩니다[S43]. 2025년 5월판 「클라우드 네이티브 구축·운영 가이드」는 기술 현황 문항을 일부 추가했지만, 체크리스트가 제시하는 도입 필요 단계는 최종 결과가 아니며 상세설계에서 정한다는 단서를 붙였습니다[S45]. 즉 애플리케이션의 기술 스택과 버전, 코드 종속성을 실제로 조사하는 일은 발주 이후의 사업자에게 맡겨져 있으며, 사전 진단은 그 조사를 본사업 전에 수행합니다.

체크리스트 13개 대분류의 구성, 정적 분석이 내는 지표와 그 한계, 두 결과를 합쳐 전환 가능 여부를 판정하는 절차를 차례로 확인하면, IT 담당자는 진단에 제출할 자료의 범위를, 의사결정권자는 판정 근거가 어느 자료에서 나오는지 알 수 있습니다.

3.1 고객 기관이 작성하는 현황 체크리스트 13개 대분류의 구성

사전 진단은 전환대상기관이 현황 체크리스트를 작성하는 단계에서 시작하며, 체크리스트는 애플리케이션 개요에서 애플리케이션 상세까지 13개 대분류로 구성됩니다. 13개 대분류는 묻는 내용에 따라 인프라 영역, 애플리케이션 영역, 규모 영역으로 묶을 수 있습니다. 인프라 영역은 서버 구성, 연계 시스템·개발 도구, WAS 상세, 성능, 안정성·장애 대응의 다섯 분류이고 컨테이너 플랫폼의 자원과 구성을 정하는 데 쓰입니다. 애플리케이션 영역은 개요, WAS 종속성, 프레임워크, 업무 소프트웨어, 기반·보안 소프트웨어, 현행 운영 이슈, 추가 요청사항의 일곱 분류이며 코드 수정 범위와 예외 판정의 근거가 됩니다. 규모 영역은 13번째 분류인 애플리케이션 상세 하나로, 공수와 기간을 산정하는 입력값을 제공합니다.

체크리스트 작성은 본사업 계약에서 전환대상기관이 지는 정보 제공 의무를 미리 이행하는 일이기도 합니다. 행정·공공기관 클라우드 네이티브 전환지원 사업의 계약특수조건은 전환대상기관이 클라우드 네이티브 전환 체크리스트를 최종 확인하고 OS, WEB/WAS, DBMS 등의 전환사항과 데이터 이관, 보안정책 정보를 제공하도록 정하고 있습니다[S42]. 사전 진단 단계에서 체크리스트를 작성해 두면 본사업에서 제공해야 할 정보가 미리 준비됩니다. 작성 과정에서는 기관이 답하지 못하는 문항도 드러나는데, 답하지 못한 문항의 목록은 본사업 전에 보완해야 할 현황 자료의 목록으로 그대로 쓸 수 있습니다.

13개 대분류는 담당자 한 명이 모두 작성하기 어려우므로 영역별로 작성자를 나눕니다. 서버 구성과 성능 문항은 인프라 운영 담당자가, WAS 종속성과 프레임워크 문항은 개발 담당자나 유지보수 업체가 답해야 정확합니다. 진단 일정을 정할 때는 영역별 작성 담당자를 먼저 지정하고, 작성된 내용은 인터뷰에서 다시 확인하는 순서로 진행합니다.

13개 대분류는 세 영역으로 묶이고, 영역마다 쓰임이 다르다

현황 체크리스트 13개 대분류의 세 영역 구성과 각 영역이 근거가 되는 결정

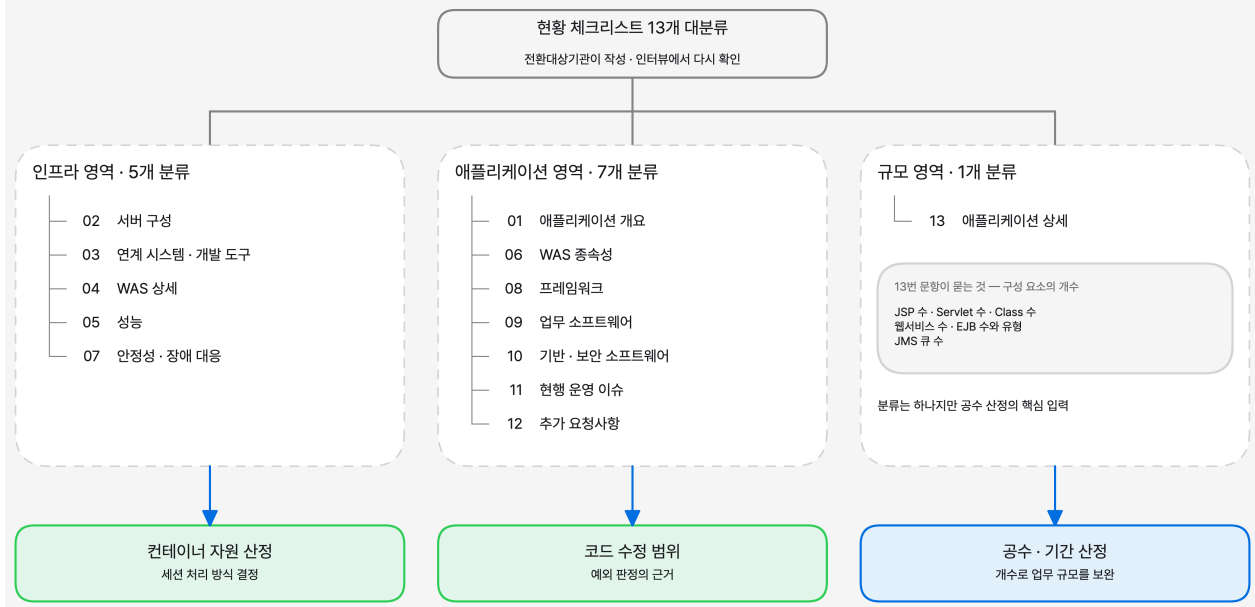


그림. 현황 체크리스트 13개 대분류의 세 영역 구성과 각 영역의 쓰임

3.1.1 서버 구성·WAS 상세·성능을 묻는 인프라 영역 문항

서버 구성 문항은 컨테이너 자원 산정의 출발값이다

서버 구성 문항은 웹 서버, WAS, DB 서버별로 설치된 소프트웨어와 CPU, 메모리, OS, 수량을 묻고 스토리지와 로드밸런서 구성을 함께 묻는데, 이 값이 컨테이너 플랫폼에 필요한 자원을 계산하는 출발값입니다. 2025년 5월판 「클라우드 네이티브 구축·운영 가이드」의 산정식은 CPU를 기존 코어 수에 평균 사용률과 여유율 1.3을 곱해 구하고, Unix 장비의 코어에는 0.8을 한 번 더 곱합니다. 클라우드 네이티브 구성에서는 같은 식에 MSA가중치 1.4를 곱하며, 메모리 여유율도 1.3 대신 1.8을 적용합니다[S45]. 따라서 서버별 코어 수와 OS 종류가 빠진 체크리스트로는 이 산정식을 적용할 수 없고, 필요 자원 수량도 근거 없이 정하게 됩니다.

WAS 상세 문항은 메모리 한도와 세션 처리 방식을 정한다

WAS 상세 문항은 티어 구성, 배포된 war(웹 애플리케이션 배포 파일) 수, 힙(heap, Java 가상 머신이 객체를 저장하는 메모리 영역) 크기, 커넥션 풀, 세션 클러스터링 여부를 묻습니다. 힙 크기와 커넥션 풀 설정은 컨테이너 한 개에 할당할 메모리 한도와 DB 연결 수를 정하는 근거이고, 세션 클러스터링 문항은 WAS 인스턴스끼리 세션을 복제하는 구성인지를 확인하는 문항입니다. 컨테이너 환경에서는 세션 복제와 배포 관리를 WAS 대신 플랫폼이 맡으므로, IP를 지정해 세션을 복제하는 구성은 그대로 옮기지 못하고 세션 처리 방식을 다시 정해야 합니다. 다만 이 항목은 컨테이너 전환을 막는 사유에 해당하지 않으며 전환 과정에서 수정할 대상으로 분류합니다(6장).

성능 문항은 전환 전후 비교의 기준값이다

성능 문항이 묻는 동시 사용자 수, 평균 응답시간, 요청당 데이터 크기는 컨테이너의 복제본 수와 자원 한도를 정할 때 쓰이고, 전환 후 성능을 전환 전과 비교하는 기준값으로도 쓰입니다. 지방교육청 통합누리집을 대상으로 한 2025년 전환 사업 제안요청서는 전환 후에도 기존과 동등한 성능을 보장하도록 요구합니다[S41]. 그래서 전환 전 수치가 없는 시스템은 본사업에서 동등 성능을 증명할 기준을 따로 만들어야 합니다. 안정성·장애 대응

문항이 빈번한 장애와 성능 이슈, 모니터링 도구 사용 여부를 함께 묻는 것도 전환 전 상태를 기록해 두기 위해서입니다.

다음 표는 인프라 영역의 다섯 분류를 대표 문항, 진단에서의 쓰임으로 정리한 것입니다.

대분류	대표 문항	진단에서의 쓰임
서버 구성	웹·WAS·DB 서버별 소프트웨어, CPU·메모리·OS·수량	컨테이너 자원 산정식의 입력
연계 시스템·개발 도구	연계 서버, 형상관리, JDK, 모니터링	연계 범위와 빌드 환경 확인
WAS 상세	티어 구성, 힙 크기, 커넥션 풀, 세션 클러스터링	메모리 한도와 세션 처리 방식 결정
성능	동시 사용자, 평균 응답시간, 요청당 데이터 크기	복제본 수 산정, 전환 전후 비교 기준
안정성·장애 대응	빈번한 장애, 성능 이슈, 모니터링 도구 사용 여부	전환 때 함께 해결할 운영 문제 파악

3.1.2 WAS 종속성·프레임워크·EJB 사용 여부를 묻는 애플리케이션 영역 문항

WAS 종속성 문항은 정적 분석 결과와 대조하는 자료다

WAS 종속성 문항은 특정 WAS 제품에 종속된 JMX(Java Management Extensions, Java 관리 확장), 웹 서비스, 세션 라이브러리를 사용하는지와 세션을 공유하는지를 묻습니다. 상용 WAS가 제공하는 전용 라이브러리를 호출하는 코드는 오픈소스 WAS로 옮기면 동작하지 않으므로 표준 방식으로 바뀌어야 합니다. 정적 분석도 대상 환경이 지원하지 않는 독점 API를 Mandatory 이슈로 분류해 찾아냅니다. 이 문항의 답은 정적 분석 결과와 대조하는 데 쓰이며, 담당자가 「종속 라이브러리 없음」이라고 답했는데 분석에서 독점 API가 검출되면 그 차이를 인터뷰에서 확인합니다.

소스 보유 여부는 예외 판정에 직접 쓰인다

프레임워크 문항은 소스 보유 여부, 업무 프레임워크와 코어 프레임워크, EJB(Enterprise JavaBeans, 서버 측 업무 컴포넌트 규격)와 JMS(Java Message Service, 메시지 연계 규격) 사용 여부를 묻습니다. 소스가 없는 애플리케이션은 코드를 고칠 수 없으므로 컨테이너 전환에 필요한 수정이 불가능합니다. 2025년 5월판 가이드도 현재 버전의 소스코드를 보유하지 않은 경우와 소프트웨어 제작업체가 없는 경우를 전환 불가 판정의 예시로 들고 있으므로[S45], 소스 보유 여부는 6장에서 다루는 예외 판정에 직접 쓰이는 답입니다. 반면 EJB와 JMS 사용 여부는 예외 사유에 해당하지 않으며, 전환 작업의 공수를 늘리는 요인으로 다룹니다.

소프트웨어 목록은 제품명과 버전까지 적는다

업무 소프트웨어 문항은 제품명, 버전, 수량을 현재와 향후로 나누어 묻고, 기반·보안 소프트웨어 문항은 접근 제어, 서버보안, 백업, DB 암호화 제품을 묻습니다. 두 문항에서 나온 제품 목록은 제품별로 벤더가 컨테이너 환경을 지원하는지 확인하는 출발 자료입니다. 2025년 5월판 가이드의 전환 사업 요구수준 평가 32문항에도 적용 기술이 지나치게 오래됐는지와 소프트웨어 다수가 오픈소스로 대체하기 어려운지를 묻는 문항이 있습니다[S45]. 다만 가이드의 문항은 5점 척도로 답하는 수준 평가이므로 어느 제품의 어느 버전이 문제인지는 알려 주

지 않습니다. 체크리스트에는 Apache Tomcat 8.5, Spring Framework 4.3, 전자정부 표준프레임워크 3.6 처럼 제품명에 버전을 붙여 적어야 지원 종료 여부를 판정하고 대체 가능성을 검토할 수 있습니다.

다음 표는 애플리케이션 영역의 대분류별 대표 문항을 진단에서의 쓰임에 맞춰 정리한 것입니다.

대분류	대표 문항	진단에서의 쓰임
애플리케이션 개요	업무명, 분류(웹·배치·연계), 작성 시기, 구축 업체	진단 대상 목록과 유지보수 주체 확인
WAS 종속성	상용 WAS 종속 JMX·웹서비스·세션 라이브러리, 세션 공유	정적 분석의 독점 API 검출 결과와 대조
프레임워크	소스 보유 여부, 업무·코어 프레임워크, EJB, JMS	예외 판정(6장), 공수 가산 요인 확인
업무 소프트웨어	제품명·버전·수량(현재/향후)	벤더의 컨테이너 지원 여부 확인
기반·보안 소프트웨어	접근제어, 서버보안, 백업, DB 암호화	컨테이너 환경에서의 대체 수단 확인
현행 운영 이슈·추가 요청사항	운영 중인 문제와 기관의 요청 내용	전환 범위에 넣을 개선 과제 선별

3.1.3 JSP·Servlet·EJB 개수로 구성되는 공수 산정 핵심 입력

애플리케이션 상세 문항은 구성 요소의 개수를 묻는다

13번째 대분류인 애플리케이션 상세는 앞의 열두 분류와 달리 구성 요소의 개수를 묻는 분류이며, 문항은 JSP(JavaServer Pages, 화면을 만드는 서버 측 페이지) 수, Servlet 수, Class 수, 웹서비스 수, EJB 수와 유형, JMS 큐 수입니다. 사전 진단에서는 이 개수 정보를 공수와 기간을 산정하는 핵심 자료로 사용합니다. 개수는 소스 저장소나 배포 파일에서 셀 수 있는 값이므로 담당자의 기억에 의존하지 않고 작성할 수 있고, 진단 수행자가 정적 분석 결과의 목록과 대조해 검증할 수도 있습니다.

JSP·EJB 개수는 도구 점수가 담지 못하는 업무 규모를 보완한다

정적 분석의 총점은 이슈가 발견된 횟수에 건당 점수를 곱한 값이어서 고쳐야 할 코드의 양은 나타내지만, 고친 뒤에 검증해야 할 화면과 기능의 수는 나타내지 않습니다. 예를 들어 대학 E의 진단에서는 도구 점수가 낮게 나온 애플리케이션에 12일, 27일이 산정됐는데, 일수를 정할 때 점수에 더해 업무 크기, 사용 컴포넌트를 함께 반영했기 때문입니다. 표본 보고서 1건에서도 약 3,000개의 JSP와 다수의 EJB가 공수를 늘린 요인으로 기록됐습니다. 따라서 점수가 낮다는 이유로 전환 기간을 짧게 정하면 검증 일수가 빠진 건적이 되며, 개수 정보는 그 누락을 막는 자료입니다(점수를 일수로 환산하는 방법은 4장).

다음 표는 13개 대분류 전체를 번호, 주요 항목, 소속 영역으로 정리한 것입니다.

번호	대분류	주요 항목	영역
1	애플리케이션 개요	업무명, 분류(웹·배치·연계), 작성 시기, 구축 업체	애플리케이션
2	서버 구성	웹·WAS·DB 서버별 소프트웨어, CPU, 메모리, OS,	인프라

번호	대분류	주요 항목	영역
		수량, 스토리지, 로드밸런서	
3	연계 시스템·개발 도구	연계 서버, 모니터링, 형상 관리, JDK, 개발 도구	인프라
4	WAS 상세	티어 구성, war 수, 힙 크기, 커넥션 풀, 세션 클러스터링	인프라
5	성능	동시 사용자, 평균 응답시간, 요청당 데이터 크기	인프라
6	WAS 종속성	종속 JMX·웹서비스·세션 라이브러리, 세션 공유	애플리케이션
7	안정성·장애 대응	빈번한 장애, 성능 이슈, 모니터링 도구 사용 여부	인프라
8	프레임워크	소스 보유 여부, 업무·코어 프레임워크, EJB, JMS	애플리케이션
9	업무 소프트웨어	제품명·버전·수량(현재/향후)	애플리케이션
10	기반·보안 소프트웨어	접근제어, 서버보안, 백업, DB 암호화	애플리케이션
11	현행 운영 이슈	이슈와 내용	애플리케이션
12	추가 요청사항	요청 내용	애플리케이션
13	애플리케이션 상세	JSP·Servlet·Class·웹서비스 수, EJB 수와 유형, JMS 큐 수	규모

3.2 소스와 바이너리를 규칙으로 검사하는 정적 분석의 산출 지표

애플리케이션 현대화 분석 도구는 소스 코드 또는 EAR·WAR·JAR 형태의 바이너리(컴파일된 배포 파일)를 규칙(rule)으로 검사하는 정적 분석 도구입니다. 정적 분석은 애플리케이션을 실행하지 않고 코드와 설정 파일을 읽어 문제를 찾는 방식이므로 운영 중인 시스템에 부하를 주지 않습니다. 도구는 규칙에 해당하는 코드마다 이슈 분류와 Story Point를 붙이며, 건당 점수는 0·1·3·5·7·13 가운데 하나이고 애플리케이션의 총점은 발견 횟수에 건당 점수를 곱해 구합니다. 이 장의 지표 설명은 CNAMT(Cloud Native Application Migration Toolkit) 문서를 기준으로 합니다.

CNAMT 문서는 Story Point가 반드시 인시로 환산되는 값은 아니며 작업 간에 일관되게 적용해야 하는 상대 값이라고 밝혀 정량 지표의 범위를 한정하고 있습니다. 총점은 애플리케이션끼리 수정 난이도를 비교하는 데 쓰

는 값이고, 그 자체가 일수나 금액은 아닙니다. 규칙이 없는 항목은 검출되지 않으므로 점수가 낮다는 결과를 문제가 없다는 뜻으로 해석해서도 안 됩니다.

분석 도구는 무료로 제공되는 것도 있으며, Microsoft는 같은 계열의 오픈소스 분석 엔진을 쓰는 Java 코드 평가 도구인 Azure Migrate의 기능으로 무료 제공합니다[S29]. 진단의 가치는 검출된 이슈 가운데 이번 전환 범위에 넣을 것을 가려내고 체크리스트 답과 맞춰 판정하는 해석 작업에 있습니다.

3.2.1 Mandatory·Optional·Potential·Information 네 가지 이슈 분류

Mandatory 이슈가 공수의 하한을 정한다

Mandatory는 고치지 않으면 빌드나 실행이 되지 않는 이슈이며, 대상 환경이 지원하지 않는 독점 API를 호출하는 코드가 대표 예입니다. Mandatory 이슈는 전환 범위에서 제외할 수 없으므로 이 분류의 점수 합이 전환 공수의 하한이 됩니다. 애플리케이션 간 우선순위를 정할 때도 총점보다 Mandatory 점수 합을 먼저 봅니다.

Optional 이슈의 반영 범위는 발주자가 결정한다

Optional은 고치지 않아도 동작하지만 최적 상태가 아닌 이슈이며, CNAMT 문서는 전환 직후 일정에 반영하도록 권장합니다. 이 분류를 이번 사업 범위에 넣을지는 기술 판단만으로 정할 수 없고 예산과 일정을 결정하는 발주자가 정해야 합니다. 목표 Java 버전과 Jakarta EE 버전을 어디로 정하느냐에 따라 이번 범위에 들어오는 이슈의 수가 크게 달라집니다(4장). 진단 결과 보고서는 Optional 이슈를 반영한 경우와 반영하지 않은 경우의 차이를 발주자가 비교할 수 있게 구분해서 제시해야 합니다.

Potential 이슈는 인터뷰에서 판정한다

Potential은 필수 여부를 판단할 정보가 부족해 사람의 검토가 필요한 이슈이고, Information은 특정 파일이 존재한다는 사실을 알리는 항목입니다. Potential 이슈는 코드만으로 판정할 수 없으므로 인터뷰에서 개발 담당자에게 해당 코드의 실제 사용 여부를 확인한 뒤 Mandatory 또는 제외로 다시 분류합니다. Information 항목은 알림 성격이어서 공수 산정에는 넣지 않으며, 애플리케이션이 사용하는 기술을 파악하는 참고 자료로 씁니다.

다음 표는 네 가지 이슈 분류의 뜻을 진단에서의 처리 방법에 맞춰 정리한 것입니다.

분류	뜻	진단에서의 처리
Mandatory	고치지 않으면 빌드 또는 실행이 되지 않는 항목	전환 범위에 반드시 포함, 공수 하한 산정
Optional	동작은 하지만 최적이지 아닌 항목	반영 여부를 발주자가 결정
Potential	필수 여부를 판단할 정보가 부족한 항목	인터뷰로 확인한 뒤 재분류
Information	파일의 존재를 알리는 항목	사용 기술 파악에 참고

출처: CNAMT CLI 가이드

3.2.2 하드코딩 IP·로컬 파일 입출력·독점 API로 대표되는 컨테이너화 이슈 검출 항목

하드코딩 IP와 로컬 파일 입출력은 줄 단위로 검출한다

정적 분석이 찾아내는 컨테이너화 이슈는 하드코딩된 경로와 IP, 로컬 파일시스템 입출력, 특정 OS 의존, 독점 API 사용의 네 종류로 묶을 수 있습니다. 컨테이너는 다시 시작할 때마다 주소가 바뀌고 내부 디스크에 쓴 파일이 남지 않으므로, 서버 주소를 코드에 적어 두었거나 업로드 파일을 로컬 디스크에 저장하는 코드는 전환 뒤에 오류를 일으킵니다. 분석 도구는 이런 코드를 파일 이름과 줄 번호까지 표시하므로 검출 결과를 개발 조직에 전달할 수정 요구사항 목록으로 바로 쓸 수 있습니다.

다음 표는 검출 항목별로 컨테이너 환경에서 생기는 문제를 일반적인 수정 방향에 맞춰 정리한 것입니다.

검출 항목	컨테이너 환경에서 생기는 문제	수정 방향
하드코딩된 경로·IP	재시작 때 주소와 경로가 바뀌어 연결 실패	설정 값을 코드 밖으로 분리
로컬 파일시스템 입출력	컨테이너가 교체되면 저장한 파일이 사라짐	외부 저장소 사용으로 변경
특정 OS 의존	리눅스 기반 이미지에서 실행되지 않음	OS에 독립적인 방식으로 변경
독점 API 사용	대상 WAS에 해당 라이브러리가 없어 빌드·실행 실패	표준 API로 교체

분석 보고서의 집계 항목은 범위와 일정 판단에 쓴다

분석 결과에는 줄 단위 검출 내역과 함께 애플리케이션별 총점, 분류별·노력 수준별·패키지별 집계, 사용 기술 목록, 의존성 목록, 하드코딩 IP 목록, EJB·JPA(Java Persistence API, 데이터 저장 규격)·Spring Bean 목록이 집계 항목으로 실립니다. 패키지별 집계는 이슈가 집중된 모듈을 알려 주므로 작업을 어느 팀에 배정할지 정할 때 쓰이고, 의존성 목록은 지원이 끝난 라이브러리를 찾는 데, EJB 목록은 체크리스트 13번 분류의 EJB 수와 대조하는 데 쓰입니다. 집계 항목은 의사결정권자가 범위와 일정을 판단하는 자료이고, 줄 단위 내역은 개발 담당자가 수정 작업에 쓰는 자료입니다. 다음 표는 집계 항목별 쓰임을 정리한 것입니다.

집계 항목	진단에서의 쓰임
애플리케이션별 총점	애플리케이션 간 난이도 비교, 우선순위 결정
분류별·노력 수준별 집계	Mandatory 점수 합 산정, 범위 결정
패키지별 집계	이슈가 집중된 모듈 파악, 작업 배정
기술·의존성 목록	지원 종료 라이브러리와 현대화 대상 식별
EJB·JPA·Spring Bean 목록	체크리스트의 개수 정보와 대조

3.2.3 Java 정식 지원과 그 밖의 언어 미리보기 지원에 따른 진단 범위 차이

Java 애플리케이션은 정량 진단이 가능하다

분석 도구가 정식으로 지원하는 언어는 Java이며, CNAMT 문서에서 Python과 Node.js 분석은 기술 미리보기(Technology Preview) 단계, .NET 분석은 개발자 미리보기(Developer Preview) 단계이고 C·C++·COBOL은 지원 언어로 나오지 않습니다. 따라서 Java 애플리케이션은 앞 절의 이슈 분류와 Story

Point로 정량 진단을 할 수 있고, 그 밖의 언어로 작성된 애플리케이션은 담당 엔지니어가 코드와 빌드 환경을 직접 검토하는 수작업 진단이 됩니다. 다만 이 구분은 CNAMT 문서가 밝힌 정식 지원 범위를 그대로 옮긴 것이며, 다른 언어용 분석 수단이 없다는 뜻은 아닙니다.

공공 전환 대상에는 C·PHP·ASP가 섞여 있다

Java 외 언어로 작성된 시스템은 공공 클라우드 네이티브 전환 사업의 대상에서 반복해서 나타납니다. 2026년 클라우드 네이티브 전환 상세설계 2차 사업의 대상 시스템 한 곳은 개발 언어에 C가 포함돼 있고[S36], 2024년 상세설계 3차 사업에는 IBM의 Unix 운영체제인 AIX에서 운영하는 PHP 홈페이지가, 4차 사업에는 ASP와 PHP 5로 작성된 시스템이 들어 있습니다[S39][S40]. 이런 대상은 수작업 진단이 필요하므로 진단 방식과 기간이 Java 애플리케이션과 달라지고, 결과도 점수 대신 검토 의견 형태로 나옵니다. 진단 범위를 정할 때는 대상 시스템의 개발 언어를 먼저 확인하고, Java 외 언어의 비중에 맞춰 기간과 투입 인력을 정해야 합니다(7장).

다음 표는 언어별 지원 단계를 진단 방식에 맞춰 정리한 것입니다.

언어	CNAMT 문서의 지원 단계	진단 방식
Java	정식 지원	정적 분석에 의한 정량 진단
Python·Node.js	기술 미리보기	수작업 진단, 도구 결과는 참고
.NET	개발자 미리보기	수작업 진단, 도구 결과는 참고
C·C++·COBOL	지원 언어로 나오지 않음	수작업 진단

출처: CNAMT CLI 가이드

3.3 체크리스트 조사와 정적 분석 결과를 합쳐 전환 가능 여부를 판정하는 절차

전환 가능 여부의 판정은 체크리스트 조사와 정적 분석 결과를 애플리케이션별로 합쳐서 내립니다. 정적 분석은 코드와 설정 파일에 적힌 내용만 읽으므로 세션을 어디에 저장하는지, 소프트웨어 벤더가 컨테이너를 지원하는지 같은 사실은 알 수 없습니다. 반면 체크리스트는 담당자가 아는 범위까지만 답이 나오므로 코드 안에 남은 하드코딩이나 독점 API 호출을 빠뜨리기 쉽습니다. 두 자료는 서로 보지 못하는 부분을 보완하는 관계이고, 어느 한쪽만으로 낸 판정은 근거가 절반만 있는 판정입니다.

평가를 코드와 그 밖의 자료로 나누어 수행하는 방식은 다른 공개 지침에서도 확인할 수 있습니다. Microsoft Cloud Adoption Framework는 워크로드 평가를 아키텍처, 구성요소, 의존성, 컴플라이언스, 코드, 데이터베이스 순으로 진행하도록 안내하며 코드 평가를 그 가운데 한 단계로 둡니다[S29]. 코드 분석 점수 하나로 전환 여부를 정하는 방식은 이러한 공개 지침의 절차와도 맞지 않습니다.

판정 절차는 전환대상기관의 체크리스트 작성에서 시작하며, 진단 수행자는 인터뷰에서 답을 확인하면서 소스 또는 바이너리를 받아 정적 분석을 수행합니다. 두 결과가 나오면 진단 수행자가 항목별로 대조해 서로 맞지 않는 부분을 담당자에게 다시 확인합니다. 그 뒤 애플리케이션마다 컨테이너 전환을 기본으로 놓고 수정 항목과 공수를 정리하며, 설문에서 예외 사유가 확인된 대상만 6장의 기준으로 따로 검토합니다. 따라서 의사결정권자가 받는 판정 결과에는 도구 점수와 함께 그 점수를 보완한 설문 항목이 근거로 붙습니다.

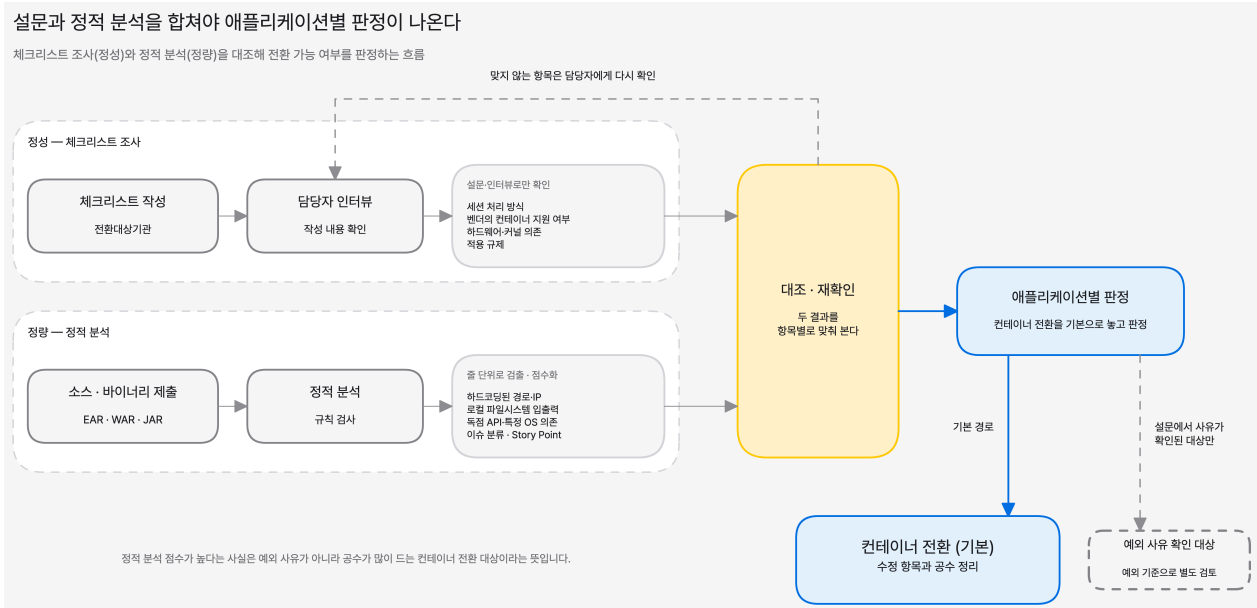


그림. 체크리스트 조사(정성)와 정적 분석(정량)을 합쳐 애플리케이션별 전환 가능 여부를 판정하는 흐름

3.3.1 설문으로만 파악되는 세션 방식·벤더 지원·하드웨어 의존·규제 항목

세션 방식과 벤더 지원 여부는 코드에 적혀 있지 않다

세션 처리 방식은 WAS 설정과 운영 구성에 따라 정해지므로 애플리케이션 코드만 읽어서는 알 수 없습니다. 같은 코드라도 WAS가 세션을 메모리에 두는지, 인스턴스끼리 복제하는지에 따라 컨테이너 전환 때 필요한 작업이 달라집니다. 소프트웨어 벤더의 컨테이너 지원 여부도 코드에는 나타나지 않습니다. 상용 제품이 컨테이너 환경을 지원하는지, 지원한다면 어느 버전부터인지는 벤더의 지원 정책에 있는 정보이므로 설문과 벤더 확인으로만 파악할 수 있습니다. 체크리스트의 WAS 상세, 업무 소프트웨어, 기반·보안 소프트웨어 문항이 이 항목의 입력이 됩니다.

하드웨어·커널 의존과 규제는 예외 사유와 일치한다

하드웨어나 커널에 의존하는 구성, 그리고 기관에 적용되는 규제도 코드에서 드러나지 않아 설문·인터뷰로 확인합니다. Google Cloud의 Migrate to Containers 계획 가이드는 특수 커널 드라이버를 쓰는 워크로드, 특정 하드웨어에 의존하는 워크로드, 하드웨어 ID에 묶인 라이선스를 쓰는 소프트웨어를 컨테이너에 맞지 않는 대상으로 들고, 웹·애플리케이션 서버와 Tomcat 같은 미들웨어는 맞는 대상으로 듭니다[S05]. 이 항목들은 6장에서 다루는 가상화 전환의 예외 사유와 일치하므로, 예외 판정은 설문 없이는 성립하지 않습니다. 반면 정적 분석 점수가 높다는 사실은 예외 사유가 되지 않으며, 점수가 높은 애플리케이션은 공수가 많이 드는 컨테이너 전환 대상으로 분류합니다.

다음 표는 확인 항목별로 정적 분석이 맞는 범위를 설문·인터뷰가 맞는 범위에 견주어 정리한 것입니다.

확인 항목	정적 분석	설문·인터뷰
하드코딩된 경로·IP	줄 단위로 검출	검출 결과의 실제 사용 여부 확인
로컬 파일시스템 입출력	줄 단위로 검출	저장 파일의 용도와 보존 기간 확인
독점 API·특정 OS 의존	줄 단위로 검출	대체 가능 여부 확인

확인 항목	정적 분석	설문·인터뷰
세션 처리 방식	확인되지 않음	WAS 상세 문항과 인터뷰로 확인
벤더의 컨테이너 지원 여부	확인되지 않음	소프트웨어 목록과 벤더 확인
하드웨어·커널 의존	확인되지 않음	설치 장치와 모듈 목록 확인
적용 규제	확인되지 않음	적용 규정과 보안정책 확인

3.3.2 Java 메이저 버전과 업데이트 번호까지 수집해야 하는 이유

JDK 8은 8u191부터 컨테이너 자원 한도를 기본으로 인식한다

JVM이 컨테이너의 자원 한도를 인식하는지는 메이저 버전보다 세부적인 업데이트 번호에서 달라집니다. 컨테이너 지원을 기본으로 활성화하는 변경은 JDK 10에 들어갔고, JDK 8 계열에는 8u191에 반영됐습니다[S07]. 같은 「Java 8」이라도 8u191보다 낮은 업데이트는 컨테이너에 설정한 메모리 한도를 기본으로 읽지 않고 호스트의 메모리를 기준으로 힙 크기를 정합니다. 이 경우 JVM이 한도를 넘는 메모리를 사용해 컨테이너가 강제 종료될 수 있습니다(5장).

Java 11도 11.0.16부터 cgroup v2를 인식한다

cgroup(control group)은 리눅스 커널이 프로세스 묶음의 CPU와 메모리 사용량을 제한하는 기능이고, 컨테이너의 자원 한도는 이 기능으로 구현됩니다. cgroup에는 v1과 v2가 있는데 JVM이 v2를 인식하는 변경은 JDK 15에 들어갔고, 이전 계열에는 11.0.16과 8u372에 반영됐습니다[S09]. 따라서 Java 11이라는 답만으로는 판정할 수 없고, cgroup v2를 쓰는 호스트에서는 11.0.16 이상인지, JDK 8 계열이면 8u372 이상인지를 확인해야 합니다.

버전 정보는 업데이트·패치 수준까지 적는다

체크리스트의 버전 문항에는 업데이트 번호가 들어간 전체 버전 문자열을 적으며, JDK 8 계열은 「1.8.0_191」, JDK 11 계열은 「11.0.16」과 같은 형식입니다. OS도 같은 원칙으로 배포판 이름에 버전을 붙여 적습니다. 배포판 공급자는 이미지와 호스트의 메이저 버전 차이에 지원 한계를 두므로, 버전이 있어야 기존 환경을 이미지로 옮길 수 있는지 판정할 수 있습니다. 다음 표는 Java, WAS, 프레임워크, OS별로 수집해야 할 버전 정보를 정리한 것입니다.

대상	수집 항목	작성 예	판정에 쓰이는 내용
Java	배포판, 메이저 버전, 업데이트 번호	OpenJDK 1.8.0_191, 11.0.16	컨테이너 자원 인식 여부, 지원 종료 여부
WAS	제품명, 버전, 패치 수준	Apache Tomcat 8.5	지원 종료 여부, 전환 대상 WAS 결정
프레임워크	제품명, 버전	Spring Framework 4.3.6, 전자정부 표준프레임워크 3.6	목표 Java 버전과의 호환 여부

대상	수집 항목	작성 예	판정에 쓰이는 내용
OS	배포판, 버전	상용 리눅스 6.10, CentOS 7.7	이미지와 호스트 간 지원 범위

3.3.3 분석 불가 판정이 나오는 산출물 형태와 보완 요청 기준

분석 불가의 원인은 제출물의 형태다

진단 사례에서 정적 분석이 분석 불가로 끝난 원인은 운영체제의 종류보다 기관이 제출한 산출물의 형태에 있었습니다. 대학 E의 진단에서는 제출물 2건이 분석 불가로 판정됐는데, 제출된 파일이 타사 라이브러리 JAR이거나 업무용이 아닌 WAR였습니다. 두 경우 모두 기관이 개발한 업무 코드가 제출물에 들어 있지 않아 규칙을 적용할 대상이 없었습니다. 즉 분석 불가는 판정에 필요한 자료가 도착하지 않았다는 뜻이며, 애플리케이션을 전환할 수 없다는 판정과는 다릅니다.

AIX의 Java 애플리케이션은 분석과 판정이 가능하다

같은 진단에서 AIX에서 운영하던 Java 애플리케이션은 분석과 판정이 모두 이뤄졌습니다. Java 가상 머신 명세는 JVM을 하드웨어와 운영체제로부터의 독립성을 담당하는 구성요소로 정의합니다[S20]. 그래서 Unix에서 운영하던 Java 애플리케이션이라도 소스나 업무용 WAR가 있으면 운영 장비와 다른 분석 환경에서 검사할 수 있습니다. 따라서 Unix 장비에서 운영한다는 사실만으로 진단 대상에서 제외하거나 예외로 분류할 필요가 없습니다. Unix의 Java 애플리케이션은 리눅스 전환을 검증한 뒤 컨테이너로 옮기는 대상입니다(6장).

제출물 기준은 진단 착수 전에 합의한다

분석 불가 판정은 제출물 기준을 진단 착수 전에 합의하면 줄일 수 있으므로, 진단 수행자는 착수 전에 애플리케이션별로 소스 코드 또는 업무 코드가 들어 있는 WAR·EAR를 요청합니다. 기관은 대상 애플리케이션의 반입과 분석 결과의 반출 권한을 미리 승인해 둡니다. 분석 불가가 나온 애플리케이션에는 부족한 제출물을 명시한 보완 요청을 보내며, 보완이 되지 않은 대상은 판정을 보류한 것으로 기록해 전환 가능 대상과 구분합니다. 진단 결과 보고서의 대상 목록에는 판정 결과 옆에 제출물 형태를 함께 기록합니다.

다음 표는 제출물 형태별 분석 가능 여부를 보완 요청 내용에 맞춰 정리한 것입니다.

제출물 형태	분석 가능 여부	보완 요청
업무 소스 코드	가능	없음
업무 코드가 들어 있는 WAR·EAR	가능	없음
타사 라이브러리 JAR만 제출	불가	업무 소스 또는 업무용 WAR 요청
업무용이 아닌 WAR	불가	운영 중인 업무용 WAR 요청
Java 외 언어의 소스	정량 분석 불가	수작업 진단으로 전환, 빌드 환경 정보 요청

4장. 난이도와 공수 산정: Story Point에서 일수까지

3장의 정적 분석이 내는 지표 가운데 공수 산정의 출발값은 Story Point입니다. Story Point는 애플리케이션 현대화 분석 도구가 소스 코드와 배포 파일에서 찾아낸 이슈마다 수정에 드는 노력을 0점에서 13점까지 매긴 점수입니다. 사전 진단은 이 점수를 컨테이너 전환 난이도의 정량 근거로 쓰고, 체크리스트와 인터뷰로 얻은 규모 정보를 더해 애플리케이션별 작업 일수를 산정합니다. 다만 Story Point를 일수나 M/M(Man-Month, 한 사람이 한 달 동안 수행하는 작업량), 기능점으로 바꾸는 공식 계수는 도구 문서에도 공공 기준에도 없습니다. 점수를 곧바로 사업비로 읽으면 견적이 실제 작업량과 어긋나고, 그 차이는 본사업 계약 이후에야 드러납니다.

공공 소프트웨어 사업의 예산은 SW사업 대가산정 가이드가 정한 경로로 편성하므로, 사전 진단이 내는 숫자가 그 경로와 어떻게 이어지는지를 발주 전에 정해 두어야 합니다[S25]. Story Point 등급의 뜻, 점수를 일수로 바꾸는 환산 관행과 한계, 진단 사례 6건에서 확인되는 점수와 일수의 관계, 대가 산정 경로와의 연결 방식을 차례로 확인하면, 의사결정권자는 진단 수치를 예산에 반영할 범위를 정할 수 있고 정책결정권자는 대가 산정 경로를 고르는 근거를 얻을 수 있습니다.

4.1 이슈 한 건마다 0점에서 13점까지 매기는 Story Point 여섯 등급

Story Point 등급은 Information, Trivial, Complex, Redesign, Rearchitecture, Unknown 여섯 가지이고, 분석 도구는 이슈 한 건마다 이 가운데 하나를 매깁니다. 애플리케이션의 총점은 이슈별 발견 횟수에 건당 점수를 곱해 더한 값이므로, 같은 총점이라도 어떤 등급이 몇 건 모여 만들어졌는지에 따라 작업의 성격이 달라집니다. 따라서 진단 보고서를 받은 의사결정권자가 먼저 확인할 값은 총점보다 등급 분포이며, 그중에서도 재설계에 해당하는 5점 이상 등급의 비중이 사업을 전환으로 볼지 재개발로 볼지를 정합니다.

4.1.1 Information 0점에서 Unknown 13점까지 여섯 등급의 뜻

Story Point 등급은 이슈 하나를 고치는 데 필요한 노력 수준을 나타내며, 점수는 0, 1, 3, 5, 7, 13의 여섯 값만 씁니다. 점수 간격이 뒤로 갈수록 벌어지는 까닭은 해결책이 불확실할수록 작업량의 편차가 커지기 때문입니다. 아래 표는 CNAMT 공식 문서가 정의한 등급을 옮긴 것으로, 등급 이름은 보고서 화면에 표시되는 영문 그대로 두었습니다.

노력 수준	점수	뜻
Information	0	우선순위가 낮거나 없는 정보성 알림
Trivial	1	단순 변경, 인터페이스 변경이 거의 없는 라이브러리 교체
Complex	3	복잡하지만 문서화된 해결책이 있는 변경
Redesign	5	재설계 또는 라이브러리 전면 교체
Rearchitecture	7	구성요소나 하위 시스템의 전면 재설계

노력 수준	점수	뜻
Unknown	13	해결책이 알려지지 않아 전체 재작성 가능성이 있는 항목

Trivial·Complex 등급의 작업 성격

Trivial 1점과 Complex 3점은 고치는 방법이 이미 알려진 이슈에 매겨집니다. 예를 들어 설정 파일에 고정된 경로를 환경 변수로 옮기거나 호환되는 라이브러리로 교체하는 작업은 건수가 많아도 한 건의 처리 방식이 같으므로 일정을 예측하기 쉽습니다. 이 두 등급이 총점의 대부분을 차지하는 애플리케이션은 총점이 수백 점이어도 기존 설계를 유지한 채 컨테이너 전환을 진행할 수 있습니다.

5점 이상 등급과 재개발 범위 판단

Redesign 5점, Rearchitecture 7점, Unknown 13점은 코드 몇 줄의 수정으로 끝나지 않고 설계 변경을 요구하는 이슈입니다. 이 세 등급의 이슈 수는 해당 애플리케이션을 전환 범위에서 처리할지 재개발 범위로 넘길지를 판단하는 근거가 됩니다. Unknown 등급은 해결책이 알려지지 않았다는 뜻이므로, 본사업 일정에 넣기 전에 기술 검증을 먼저 수행해 실제 작업량을 확인해야 합니다.

등급과 함께 읽어야 하는 축이 이슈 분류입니다. 분석 도구는 이슈를 Mandatory, Optional, Potential, Information 네 가지로 나누는데, Mandatory는 고치지 않으면 빌드나 실행이 되지 않는 항목이고 Optional은 동작은 하지만 전환 직후 개선을 권장하는 항목입니다. 즉 등급은 수정에 드는 노력의 크기를, 분류는 수정이 필요한 정도를 나타내므로 두 축을 겹쳐 보아야 이번 사업에서 반드시 처리할 작업량이 나옵니다.

4.1.2 발견 횟수와 건당 점수를 곱하는 총점 산식과 집계 단위

애플리케이션의 Story Point 총점은 이슈 종류마다 발견 횟수에 건당 점수를 곱한 뒤 모두 더한 값입니다. 발견 횟수는 같은 이슈가 코드에서 확인된 위치의 수를 뜻하므로, 한 가지 이슈가 여러 파일에 반복되면 그만큼 점수가 쌓입니다. 아래는 이슈가 세 종류인 가상의 애플리케이션으로 산식을 한 번 계산한 예입니다.

이슈	등급(건당 점수)	발견 횟수	소계
설정 파일의 고정 경로	Trivial (1)	40	40
지원이 끝난 라이브러리 호출	Complex (3)	10	30
특정 WAS 전용 기능 사용	Redesign (5)	4	20
총점			90

이 애플리케이션의 총점은 90점이지만, 그중 20점은 설계를 바꿔야 하는 네 건에서 나왔습니다. 반면 Trivial 이슈만 90건 발견된 다른 애플리케이션도 총점은 똑같이 90점입니다. 두 애플리케이션은 총점이 같아도 필요한 인력의 숙련도와 검증 범위가 다르므로, 총점 하나만 비교해서는 어느 쪽이 더 어려운 대상인지 판단할 수 없습니다.

집계 단위 네 가지의 용도

분석 보고서는 총점을 애플리케이션별, 분류별, 노력 수준별, 패키지별로 나누어 집계합니다. 애플리케이션별 총점은 여러 대상의 전환 순서를 정할 때 쓰고, 분류별 집계는 Mandatory 점수처럼 반드시 처리할 작업량을 가려낼 때 씁니다. 노력 수준별 집계는 5점 이상 등급의 비중을 보여 주어 사업 성격을 판단하게 하고, 패키지별 집계는 수정이 몰린 업무 모듈과 담당 조직을 찾는 데 씁니다.

총점은 서로 다른 애플리케이션의 난이도를 같은 기준으로 견주는 상대 지표로 읽는 것이 적절합니다. 보고서에는 점수 외에도 사용 기술 목록, 의존 라이브러리, 코드에 고정된 IP 주소, EJB와 Spring Bean 목록이 함께 실립니다. 점수가 높게 나온 이유를 이 목록에서 확인한 다음에 공수 산정으로 넘어가야 점수와 실제 작업 내용이 서로 맞습니다.

이슈 분류별 발견 횟수와 건당 점수를 집계한 보고서 표의 예시는 부록의 함께 읽을 자료 [R1]에 실은 발표자료에서 볼 수 있습니다.

4.2 Story Point를 작업 일수로 바꾸는 환산 관행과 그 한계

Story Point를 작업 일수로 바꾸는 방법은 수행 조직이 실적에 맞춰 써 온 관행이며, 도구 공식 문서나 공공 기준이 정한 계수가 아닙니다. 사전 진단에서는 Story Point 한 점을 3시간으로 환산하는 것을 기준으로 삼고 애플리케이션마다 기본 작업 5일을 더합니다. 수행자가 대상 스택과 수정 패턴을 보고 경험에 따라 조정된 값을 기준값 대신 넣기도 하며, 이와 별도로 인터뷰 점수로 상·중·하 난이도를 정합니다. 그러나 도구 공식 문서는 Story Point가 반드시 인시로 환산되지는 않는다고 적고 있으므로, 환산 결과는 단일 값이 아닌 범위로 제시해야 합니다. 단일 계수로 계산한 일수를 예산 근거로 쓰면 계수의 오차가 사업비에 그대로 반영됩니다.

4.2.1 Story Point 한 점당 3시간 기준 환산과 기본 작업 5일 가산 방식

사전 진단에서는 Story Point 한 점을 3시간으로 환산하는 것을 기준으로 삼습니다. 다만 이 기준값을 모든 대상에 기계적으로 곱하지는 않습니다. 수행자는 대상 스택, 같은 수정이 반복되는 정도, 작업자의 숙련도를 보고 경험에 따라 값을 조정하며, 해당 전환 경로를 여러 번 수행한 팀은 한 점을 1시간에 가깝게 잡기도 합니다. 조정에는 수행자의 주관적 판단이 들어가므로 조정된 값과 그 근거를 산정 결과에 함께 적습니다. 3시간이라는 기준값은 수행 조직의 관행이며 분석 도구 제조사나 공공 기관이 정한 공식 기준이 아닙니다. 환산값을 인용할 때는 어느 조직이 어떤 대상에 적용한 값인지를 함께 밝혀야 합니다.

기본 작업 5일의 구성

환산한 시간에는 애플리케이션마다 기본 작업 5일을 더합니다. 기본 작업은 컨테이너 실행 환경에 제품을 설치하고 구성하는 일, 전환한 애플리케이션을 테스트하는 일, 담당자와 협의하는 일로 이루어지며 코드 수정량과 관계없이 발생합니다. 기본 작업이 있으므로 범위에 남은 점수가 0에 가까운 애플리케이션도 공수는 0이 되지 않고 5일에서 시작합니다. 운영 환경 구성과 성능 테스트는 이 일수에 포함하지 않고 별도로 산정합니다.

구성 요소	내용	비고
환산 단위	Story Point 한 점당 3시간 기준	수행자가 대상 조건을 보고 경험에 따라 조정하기도 하는 수행 조직 관행
환산 대상	이번 전환 범위에 남긴 점수	범위에서 제외한 점수는 환산하지 않음
가산 항목	애플리케이션당 기본 작업 5일	설치·구성·테스트·협의

구성 요소	내용	비고
별도 산정	운영 환경 구성, 성능 테스트	환산 일수에 포함하지 않음

NIA 가이드의 IaaS 전환 공수 예시

공공 부문에서 이 관행과 견줄 수 있는 공개 수치는 「클라우드 네이티브 구축·운영 가이드」(2025년 5월) 부록의 설계기준 예시표입니다[S45]. 이 표는 작업 항목마다 서버당 투입 일수(M/D)를 제시하는데, WAR·EAR 배포와 하드코딩 수정은 각각 9 M/D, JDK 버전 변경에 따른 프레임워크 버전 상향은 17 M/D입니다. 다만 이 표는 IaaS 전환용이고 컨테이너 전환용 대응표는 가이드에 없으므로, 두 수치를 같은 단위로 놓고 비교할 수는 없습니다. JDK 버전 변경을 동반하는 작업이 가장 무겁다는 방향만 양쪽에서 같게 확인됩니다.

일수를 M/M으로 옮길 때는 월 근무일 기준을 하나로 통일해야 합니다. 이 백서는 월 20일을 기준으로 하며, 예를 들어 15일은 0.75 M/M, 5일은 0.25 M/M으로 적습니다. 기준이 다른 M/M 수치가 한 문서에 섞이면 같은 일수가 서로 다른 값으로 표시되어 사업 간 비교가 어긋나므로, 가능하면 일수를 함께 적는 편이 안전합니다.

4.2.2 도구 공식 문서가 인시 환산을 유보한 까닭과 범위 제시 원칙

CNAMT 공식 문서는 Story Point가 반드시 인시로 환산되지는 않으며 작업 간에 값이 일관되어야 한다고 설명합니다. 도구가 보장하는 것은 이슈 사이의 상대 크기이고, 한 점이 몇 시간인지는 도구의 책임 범위 밖에 있습니다. 한 점을 시간으로 정한 제조사 공식 자료는 확인되지 않습니다.

한 점의 작업량이 달라지는 조건

도구가 인시 환산을 유보한 까닭은 팀의 숙련도와 대상 스택에 따라 한 점에 드는 시간이 달라지기 때문입니다. 같은 Complex 3점 이슈라도 해당 WAS에서 다른 WAS로 옮긴 경험이 있는 팀은 정해진 절차대로 처리하지만, 처음 다루는 팀은 해결책을 찾고 검증하는 시간이 추가로 듭니다. 테스트 자동화 수준과 업무 담당자의 협조 정도처럼 코드 밖의 조건도 작업 시간을 바꿉니다. 따라서 한 조직의 계수를 다른 조직이나 다른 스택에 그대로 적용하면 오차가 생깁니다.

아래 표는 이번 범위에 남은 점수가 120점인 가상의 애플리케이션을 두 방식으로 제시한 예입니다. 단일 값은 계산이 간단하지만 계수가 틀렸을 때의 차이를 독자가 알 수 없습니다. 반면 범위 제시는 하한과 상한의 조건을 함께 보여 주므로, 예산 담당자가 어느 쪽에 가깝게 편성할지를 판단할 수 있습니다.

제시 방식	계산	결과	함께 밝히는 정보
단일 값	120점 × 3시간(기준값) + 기본 5일	50일	없음
범위	120점 × 1~3시간(조정 범위) + 기본 5일	20~50일	조정 근거, 표본 수, 대상 스택, 목표 버전

공수 범위와 함께 밝힐 조건

공수는 범위로 제시하고, 그 범위가 어떤 실적에서 나왔는지를 표본 수와 대상 스택으로 밝히는 것이 원칙입니다. 예를 들어 「Tomcat과 Spring 4 기반 애플리케이션 3건의 실적으로 보정한 값」처럼 조건을 적으면 독자가 자기 시스템에 적용할 수 있는지를 판단할 수 있습니다. 목표 Java 버전과 범위에서 제외한 점수도 함께 적어야 하는데, 이 두 조건이 공수에 미치는 영향은 4.3절의 사례에서 확인됩니다.

4.2.3 인터뷰 점수로 정하는 상·중·하 난이도 구간과 전환 소요 기간

사전 진단은 분석 도구의 Story Point와 별개로 인터뷰 기반 진단 시트의 점수로 애플리케이션의 난이도를 상·중·하로 나눕니다. 진단 시트 점수는 인터뷰와 체크리스트 응답으로 매기는 0점에서 100점 사이의 값이며, 이름은 같은 점수이지만 Story Point와는 척도가 다릅니다. 아래 구간은 사전 진단 수행 조직이 쓰는 기준이고 공공 기준으로 정해진 값은 아닙니다.

난이도	진단 시트 점수	주요 작업	전환 소요
하	0~10	제품 설치, 전환, 검증	1개월 미만
중	11~30	제품 설치, 전환 계획 수립, 전환, 연동, 검증	1개월 이상 2개월 미만
상	31~100	제품 설치, 전환 계획 수립, 전환, 연동, 검증	2개월 이상

난이도 하는 설치, 전환, 검증만으로 끝나는 대상이고, 중과 상에는 전환 계획 수립과 연동 작업이 추가됩니다. 즉 구간을 나누는 기준은 코드 수정량보다 연계 시스템의 수와 사전 계획의 필요 여부에 가깝습니다. Story Point가 낮은 애플리케이션도 연계 대상이 많으면 난이도 중으로 분류될 수 있습니다.

NIA 가이드의 상·중·하 표기와 비교

「클라우드 네이티브 구축·운영 가이드」의 IaaS 전환 예시표도 난이도를 상·중·하로 표기합니다[S45]. 이 표에서 운영체제를 Unix에서 리눅스로 바꾸는 작업은 상, 리눅스에서 리눅스로 옮기는 작업은 중, WEB·WAS 라이선스를 재활용하는 작업은 하입니다. 다만 가이드는 작업 항목 단위로 난이도를 매기고 사전 진단은 애플리케이션 단위로 매기므로, 두 표기를 나란히 놓을 때는 단위의 차이를 함께 밝혀야 합니다.

정량 점수와 정성 난이도를 함께 제시하는 까닭은 각 방법이 보지 못하는 항목을 다른 방법이 보완하기 때문입니다. 정적 분석은 코드에 고정된 경로와 IP 주소, 특정 운영체제 의존, WAS 전용 기능처럼 코드에 드러난 항목을 찾습니다. 반면 세션 처리 방식, 상용 소프트웨어 벤더의 컨테이너 지원 여부, 하드웨어와 커널 의존은 인터뷰와 체크리스트로만 파악됩니다. 보고서에는 Story Point와 난이도 구간을 함께 싣고, 두 값이 어긋나는 대상은 그 이유를 따로 적습니다.

4.3 익명 진단 사례 6건에서 확인되는 점수와 일수의 관계

2020년부터 2025년 사이에 수행된 사전 진단 사례 6건은 Story Point와 산정 일수가 비례하지 않는다는 점을 수치로 보여 줍니다. 총점이 900점을 넘는 애플리케이션에 5일이 산정된 사례가 있고, 총점이 한 자릿수인 애플리케이션에 27일이 산정된 사례도 있습니다. 이 차이는 목표 Java 버전에 따라 범위에서 제외되는 점수, 업무 규모, 운영체제 전환 여부에서 나옵니다. 따라서 Story Point만으로는 견적을 낼 수 없으며, 발주자가 목표 버전과 전환 범위를 먼저 정해야 공수가 정해집니다.

4.3.1 목표 Java·Jakarta EE 버전에 따라 달라지는 생략 가능 점수

진단 사례 다수에서 Story Point 총점의 대부분은 이번 전환 범위에서 생략할 수 있는 점수로 분류됐습니다. 아래 표는 사례 6건의 기존 스택, 애플리케이션별 총점과 생략 가능 점수, 산정 일수, 난이도를 기관명 없이 정리한

것입니다. 표본 보고서는 총점이 본문에 실려 있지 않아 일수만 적었고, 대학 E의 점수는 분석 결과의 추출 상태가 좋지 않아 추정값입니다.

사례	기존 스택	애플리케이션별 총점 (생략 가능)	산정 일수	난이도
표본 보고서	CentOS 6, 오픈소스 기반 WAS·Tomcat, Java 1.6	미기재	10일, 5일	미표기
공공기관 A	상용 리눅스 6.10, JEUS 6, JDK 1.7, 전자정부 표준프레임 워크 3.6	836(836), 861(861)	2개 합계 10일	미표기
대학 B	상용 리눅스 6.7, 오픈소스 기반 WAS, JSP	150(150), 176(176)	각 5일	하
유통사 C	상용 리눅스 8.4, Spring 4.3.6	52, 60, 61(전량)	각 5일	하
공공기관 D	CentOS 6.8, Tomcat, Spring 4.0.9	937(920)	5일	하
대학 E	CentOS·AIX 6.1, JEUS 8	18, 9, 4, 4(추정)	5일, 27일, 12일, 12일	중

공공기관 A의 두 애플리케이션은 총점 836점과 861점이 모두 생략 가능 점수였고, 공공기관 D는 937점 가운데 920점이 생략 가능 점수였습니다. 생략 사유는 두 가지입니다. 하나는 Java 패키지 이름을 javax에서 jakarta로 바꾸는 작업을 이번 범위에서 제외한 것이고, 다른 하나는 OpenJDK 11에서 제거된 모듈과 관련한 항목을 제외한 것입니다. 이 사례들의 산정 일수는 그 결과 기본 작업 5일에 가까운 값으로 모였습니다.

목표 버전과 작업 대상 점수

생략 가능 점수는 목표 버전을 낮게 정했을 때만 생략할 수 있습니다. Jakarta EE 9는 2020년 12월에 패키지 이름을 javax에서 jakarta로 바꿨고, Spring Boot 3.0은 Java 17과 Spring Framework 6을 요구합니다 [S12][S13]. Tomcat은 10 버전부터 jakarta 패키지를 쓰며, 전자정부 표준프레임워크 5.0.0은 Java 17 이상을 기반으로 합니다[S15][S16]. 따라서 목표를 Java 17 이상으로 정하면 위 표에서 생략한 점수가 모두 작업 대상이 되고, 5일로 산정된 애플리케이션의 공수는 다시 계산해야 합니다.

목표 Java 버전과 Jakarta EE 버전은 발주자가 사전 진단을 시작하기 전에 정해야 하는 첫 번째 값입니다. 목표를 정하지 않은 채 받은 총점은 어느 점수까지 이번 사업에 포함되는지 알 수 없으므로 예산 근거로 쓸 수 없습니다. 구버전 Java와 WAS를 유지한 컨테이너 전환이 왜 성립하기 어려운지는 지원 종료 일정과 함께 5장에서 다룹니다.

4.3.2 한 자릿수 점수에 12~27일이 산정된 사례와 업무 규모 변수

대학 E 사례에서는 Story Point가 한 자릿수인 애플리케이션에 12일에서 27일이 산정됐습니다. 네 애플리케이션의 점수는 18점, 9점, 4점, 4점이었고 산정 일수는 차례로 5일, 27일, 12일, 12일이었습니다. 다만 이 사례의 점수는 분석 결과의 추출 상태가 좋지 않아 추정한 값이므로, 개별 수치보다 점수가 가장 높은 애플리케이션의 일수가 가장 짧았다는 방향을 읽어야 합니다.

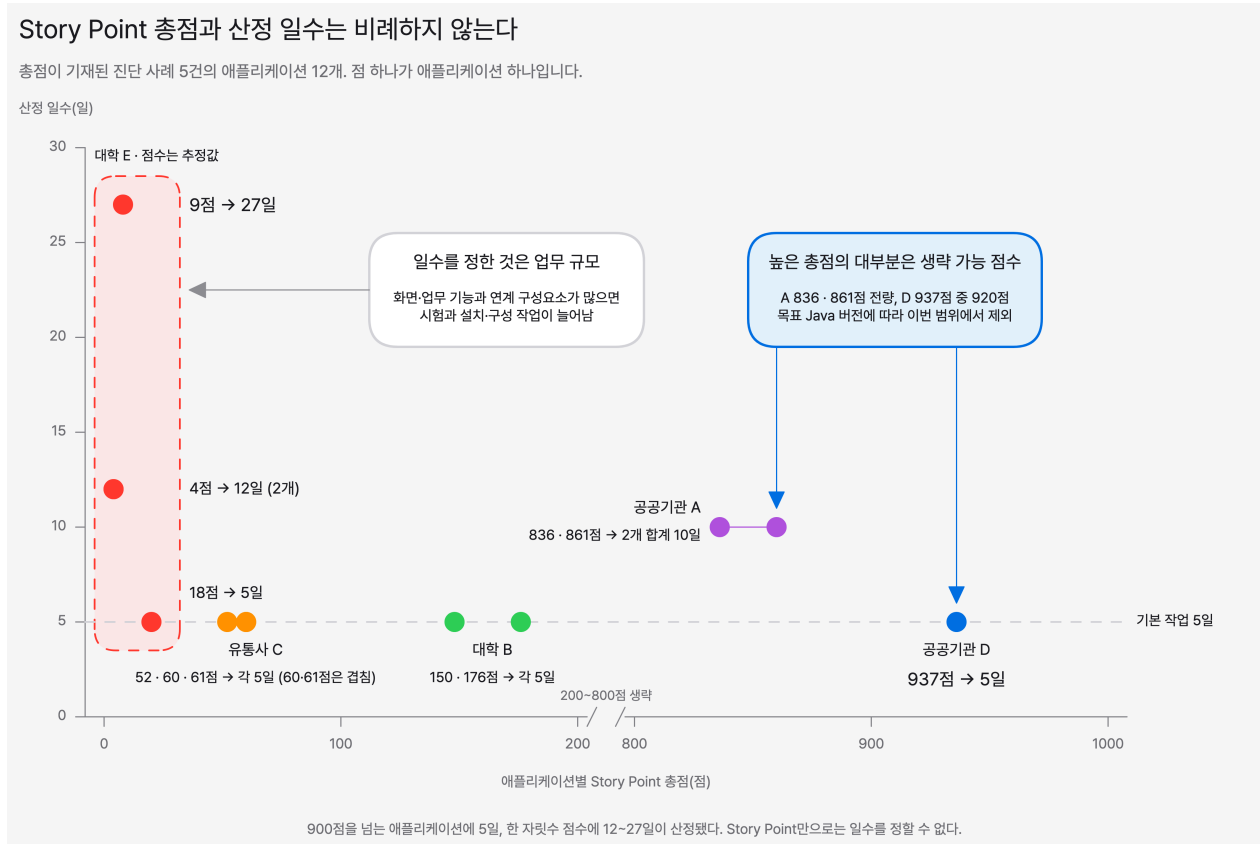


그림. 진단 사례의 Story Point 총점과 산정 일수 분포

산정 일수를 정한 업무 규모

대학 E의 일수는 Story Point보다 업무 크기와 사용 구성요소를 기준으로 정해졌습니다. 화면과 업무 기능이 많은 애플리케이션은 수정할 코드가 적어도 전환 후에 확인해야 할 시험 항목이 많고, 연계하는 구성요소가 많으면 설치와 구성 작업도 늘어납니다. 정적 분석 점수는 고쳐야 할 코드의 양을 나타낼 뿐이며, 고친 뒤에 검증해야 할 범위는 알려 주지 않습니다.

반대 방향의 사례도 같은 표에 있습니다. 공공기관 D는 총점이 937점이었지만 생략 가능 점수를 뺀 나머지가 적어 5일이 산정됐습니다. 그러므로 Story Point만으로 견적을 내면 대학 E와 같은 대상은 과소 산정되고 공공기관 D와 같은 대상은 과대 산정됩니다. 체크리스트로 수집하는 화면 수, 연계 대상, 사용 구성요소 같은 규모 정보가 정적 분석 결과와 함께 있어야 하는 이유가 이것입니다.

대학 E 사례에서는 분석을 요청한 대상 가운데 2건이 분석되지 않았습니다. 사유는 운영체제가 아닌 제출 파일의 형태였으며, 제출된 파일이 다른 회사의 라이브러리거나 업무용 배포 파일에 해당하지 않았기 때문입니다. 반면 AIX에서 운영되던 Java 애플리케이션은 분석과 난이도 판정이 모두 이루어졌습니다. 진단 대상을 정할 때는 운영체제 종류보다 업무용 소스나 배포 파일을 확보할 수 있는지를 먼저 확인해야 합니다.

4.3.3 12 M/M 사례와 5~15일 사례의 차이를 만든 전환 범위

Unix 기반 시스템을 컨테이너로 옮기는 전환 계획 사례는 보정 후 Story Point가 1,495점이었고, 모듈 수정 공수 86일에 설치·테스트·산출물 작성 68일을 더해 154일로 산정됐습니다. 이 계획은 3명이 4개월 동안 투입되는 12 M/M으로 인력을 편성했으며, 산정 일수 154일을 월 20일 기준으로 옮기면 7.7 M/M입니다. 인력 편성과 산정 일수는 서로 다른 수치이므로 아래 비교에서는 일수를 기준으로 삼습니다. 모듈 수정 공수 86일은 1,495점에 기준값 3시간을 그대로 곱한 값과 다르며, 수행자가 모듈별 수정 패턴을 보고 조정한 결과로 읽어야 합니다. 반면 리눅스에서 운영되던 Java 애플리케이션 사례들은 애플리케이션당 5일에서 15일 사이였습니다. 두 묶음의 전제 조건을 나란히 놓으면 아래와 같습니다.

전제 조건	Unix 기반 시스템 전환 계획	리눅스 기반 Java 애플리케이션 사례
기존 운영체제	Unix	상용 리눅스, CentOS
산정 단위	시스템 전체(여러 모듈)	애플리케이션 한 개
환산 대상 점수	보정 후 1,495점	생략 가능 점수를 뺀 나머지
공수 구성	모듈 수정 86일 + 설치·테스트·산출물 68일	기본 작업 5일 + 애플리케이션별 가산
산정 결과	154일, 인력 편성 12 M/M	애플리케이션당 5~15일

공수 차이를 만든 세 조건

두 사례의 차이는 운영체제 전환 여부, 목표 Java 버전, 모듈 수에서 나옵니다. Unix에서 리눅스 컨테이너로 옮기면 운영체제에 의존하는 코드와 설정을 다시 확인해야 하고, 목표 Java 버전을 높이면 4.3.1항에서 본 생략 가능 점수가 작업 대상으로 바뀝니다. 시스템 전체를 대상으로 하면 모듈마다 설치와 시험이 반복되어 공수가 누적됩니다. NIA 가이드의 IaaS 전환 예시표도 Unix에서 리눅스로 바꾸는 작업의 난이도를 상으로 표기하고, C 언어 재컴파일을 서버당 9 M/D의 별도 항목으로 두고 있습니다[S45].

전환 범위를 밝히지 않은 공수 비교는 판단 근거가 되지 못합니다. 다른 기관의 공수를 참고할 때는 기존 운영체제, 목표 Java 버전, 대상 모듈 수, 일수에 포함한 작업의 범위를 함께 확인해야 합니다. 이 장의 수치는 모두 컨테이너 전환을 전제로 산정한 값이며, 컨테이너로 옮길 수 없는 사유가 확인된 예외 대상의 처리 기준은 6장에서 다룹니다.

4.4 Story Point와 공공 소프트웨어 사업 대가 산정을 잇는 세 경로

공공 소프트웨어 사업의 사업비는 SW사업 대가산정 가이드가 정한 방식으로 산정하며, 이 가이드가 쓰는 단위는 기능점수와 투입공수입니다[S25]. 전환 사업에 적용할 수 있는 경로는 기능점수 방식, 구현과 시험 단계만 적용하는 방식, 투입공수 방식의 세 가지입니다. Story Point는 금액을 직접 계산하는 데 쓰지 않고, 세 경로 가운데 어느 것을 적용할지 고르는 근거로 연결하는 것이 타당합니다. 이렇게 하면 정책결정권자는 진단 결과를 예산 편성에 쓸 수 있는 범위와 쓸 수 없는 범위를 구분할 수 있습니다.

4.4.1 기능점수·구현 및 시험 단계 한정·투입공수로 나뉘는 대가 산정 경로

SW사업 대가산정 가이드 2024년 개정판은 소프트웨어 개발비를 기능점수에 단가를 곱해 산정하며, 2024년 판의 기능점수당 단가는 605,784원입니다[S25]. 개발 단계별 가중치는 분석 19%, 설계 24%, 구현 32%, 시

험 25%이고, 보정계수는 규모, 연계복잡성, 성능 요구수준, 운영환경 호환성, 보안성의 다섯 가지입니다. 클라우드 네이티브나 MSA를 이유로 적용하는 보정계수는 이 가이드에 없습니다. 전환 사업에 적용할 수 있는 경로를 정리하면 아래 표와 같습니다.

경로	적용 조건	산정 방식
기능점수	기능을 추가하거나 수정하는 재개발	기능점수 × 단가, 전 단계 적용
구현·시험 단계 한정	기능 변경이 없는 전환	분석·설계 단계 제외, 구현 32%와 시험 25%만 적용
투입공수	기능점수 방식이 맞지 않는 작업	투입 인력과 기간으로 산정

출처: SW사업 대가산정 가이드 2024년 개정판[S25]

기능 변경이 없는 전환의 산정 범위

대가산정 가이드는 메인프레임 기반 시스템을 웹 기반으로 다시 개발하면서 기능 변경량이 0으로 산정된 경우를 예로 듭니다. 이 경우에는 신규 개발비 기준을 적용하되 분석과 설계 단계의 대가를 제외하고 구현과 시험 단계의 개발비만 다시 산정하는 것이 적절하다고 설명합니다[S25]. 두 단계의 가중치를 더하면 57%이므로, 같은 기능 규모라도 재개발 경로와 전환 경로의 대가는 크게 달라집니다.

투입공수 방식의 적용 대상

투입공수 방식은 기능점수로 규모를 잴 수 없는 작업에 적용합니다. 가이드는 사용자에게 식별되는 기능 규모에 비해 내부 처리 복잡도가 현저히 높은 사업, 그리고 튜닝·최적화·테스트처럼 기능점수 산정이 불가능한 작업을 그 대상으로 듭니다[S25]. 따라서 컨테이너 실행 환경 구성이나 성능 테스트처럼 화면 기능과 무관한 전환 작업은 이 경로로 산정하는 편이 설명하기 쉽습니다.

이 절의 단가와 가중치는 모두 2024년 개정판 기준입니다. 대가산정 가이드는 2025년 8월에 다시 개정됐으므로, 실제 예산을 편성할 때는 해당 연도의 개정판에서 단가와 적용 조건을 확인해야 합니다[S26].

4.4.2 재설계 등급 이슈 비중으로 대가 산정 경로를 고르는 판단 기준

Story Point의 등급 분포로 대가 산정 경로를 고르는 방법은 이 백서의 제안이며 공식 기준이 아닙니다. 대가산정 가이드가 정한 것은 세 경로의 적용 조건까지이고, 어떤 애플리케이션이 기능 변경이 없는 전환에 해당하는지를 판정하는 수치 기준은 가이드에 없습니다[S25]. 사전 진단의 등급 분포는 이 판정에 쓸 수 있는 정량 자료입니다. 다만 Potential로 분류된 이슈는 필수 여부를 판단할 정보가 부족한 항목이므로, 경로를 고르기 전에 담당자 검토로 Mandatory인지 Optional인지를 먼저 정해야 합니다.

등급 분포에 따른 경로 선택

Mandatory 점수가 낮고 수정 내용이 설정과 배포 수준에 머무는 애플리케이션은 기능 변경이 없는 전환으로 보고 구현·시험 단계 한정 경로를 적용합니다. 반면 Redesign 5점, Rearchitecture 7점, Unknown 13점 이슈가 많은 애플리케이션은 설계를 바꿔야 하므로 재개발 범위로 보고 기능점수 경로를 검토합니다. 실행 환경 구성과 성능 테스트처럼 기능점수로 식별되지 않는 작업은 어느 경우나 투입공수 경로로 따로 산정합니다.

Story Point 등급 분포로 대가 산정 경로를 고르는 판단 흐름

진단 결과는 금액을 계산하지 않고, 세 경로 가운데 어느 것을 적용할지 고르는 근거로 씁니다.

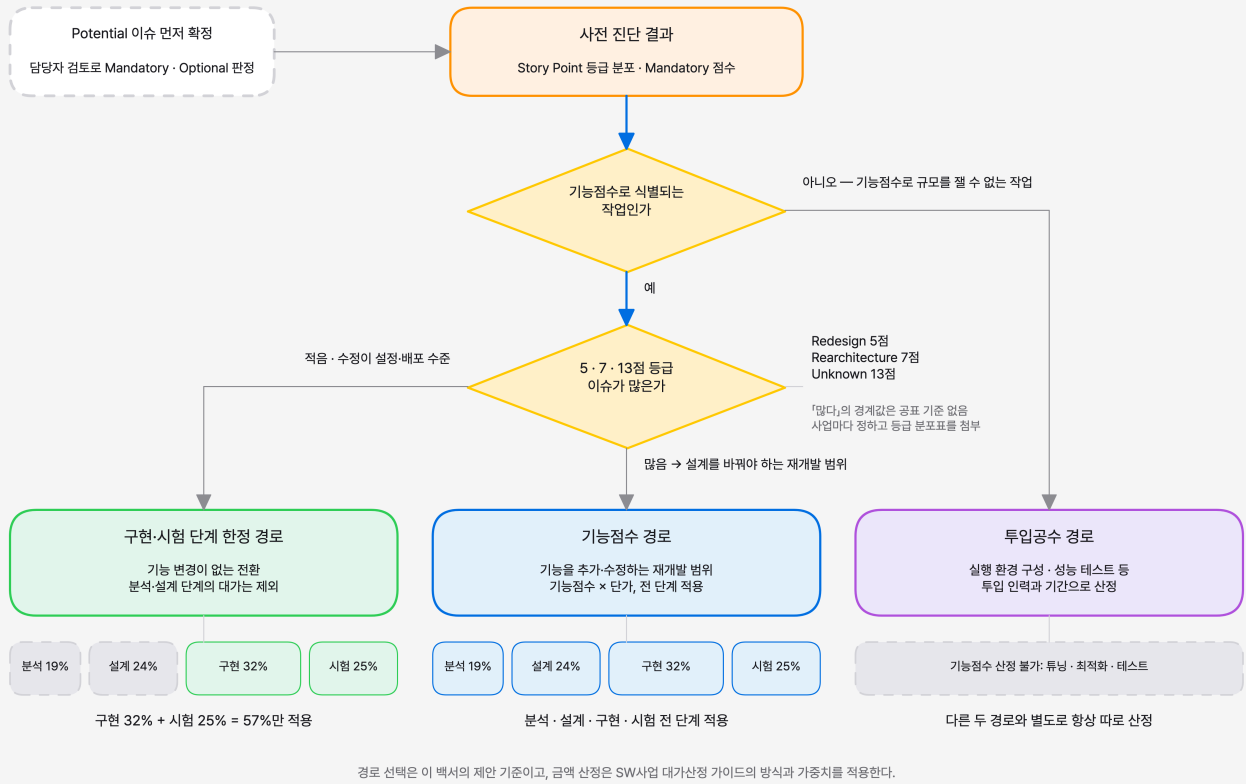


그림. Story Point 등급 분포로 대가 산정 경로를 고르는 판단 흐름

이 판단 기준에서 「많다」와 「낮다」를 가르는 경계값은 공표된 기준이 없습니다. 발주 기관은 사업마다 경계값을 정하고, 경로를 고른 근거로 애플리케이션별 등급 분포표를 예산 설명 자료에 첨부하는 방식을 권장합니다. 이렇게 하면 같은 사업 안에서 애플리케이션마다 다른 경로를 적용하더라도 선택의 이유를 수치로 설명할 수 있습니다.

4.4.3 기능점수 환산 계수가 존재하지 않는다는 사실과 서술 원칙

Story Point를 기능점수나 M/M으로 바꾸는 공식 계수는 없습니다. 분석 도구의 공식 문서는 인시 환산을 유보하고 있고, SW사업 대가산정 가이드는 기능점수와 투입공수로만 대가를 산정합니다[S25]. 클라우드 네이티브 전환 전용 대가 기준도 공표된 것이 없으며, NIA의 2021년 교육자료는 보정계수의 조정이나 기능점수의 추가라는 방향만 제시합니다[S44].

Story Point와 기능점수의 상관 연구

Story Point와 기능점수 계열 단위의 관계를 다룬 공개 연구도 환산의 근거가 되지 못합니다. 2016년에 발표된 산업 사례 연구는 Story Point가 소프트웨어의 크기를 나타내지 않으므로 생산성 판단과 프로젝트 간 비교에 쓸 수 없다고 결론지었습니다[S27]. 이 연구가 다룬 Story Point는 개발 팀이 직접 추정하는 값이어서 분석 도구의 점수와 산출 방식은 다르지만, 팀마다 한 점의 크기가 달라 환산이 성립하지 않는다는 한계는 같습니다. 두 단위의 상관을 일관되게 입증한 공개 근거는 확인되지 않습니다.

아래 표는 두 단위가 재는 대상과 쓰임을 구분한 것입니다. Story Point는 기존 코드를 고치는 노력을 재고, 기능점수는 사용자에게 제공되는 기능의 규모를 잽니다. 재는 대상이 다르므로 한쪽을 다른 쪽으로 바꾸는 계수를

가정하면 그 근거를 설명할 수 없습니다.

구분	Story Point	기능점수
재는 대상	기존 코드의 수정 노력	사용자에게 식별되는 기능 규모
산출 근거	분석 도구의 룰과 등급표	SW사업 대가산정 가이드
쓰임	난이도 비교, 전환 순서, 대가 산정 경로 선택	사업비 산정
금액 환산	공식 계수 없음	단가와 보정계수 적용

두 단위를 함께 쓰는 문서의 서술 원칙

예산 문서에서는 두 단위를 따로 산정하고 연결하는 부분만 밝히는 것이 타당합니다. 사업비는 대가산정 가이드의 경로로 산정하고, Story Point는 그 경로를 고른 이유와 애플리케이션 간 전환 순서를 설명하는 자료로 첨부합니다. 일수와 M/M은 수행 조직의 실적으로 보정한 범위임을 밝히고 표본 수와 대상 스택을 함께 적습니다. 출처가 확인되지 않은 환산 비율은 예산 문서에 실지 않아야 하며, 계수를 가정해 세운 예산은 심의에서 근거를 설명할 수 없습니다.

5장. 전환과 현대화를 함께 해야 하는 이유: 지원 종료 일정과 JVM 컨테이너 인식

4장의 진단 사례에서 생략 가능 점수의 크기를 정한 것은 목표 Java 버전이었습니다. 컨테이너 전환 대상으로 검토하는 공공·기업 시스템의 상당수는 Java 1.6~1.8, 상용 WAS, 상용 리눅스 6·CentOS 7·AIX 7.1 같은 구버전 스택에서 운영 중입니다. 2024년과 2026년 클라우드 네이티브 전환 상세설계 사업의 제안요청서에 기재된 대상 시스템에서도 JDK 1.6·1.7·1.8이 혼재하고 JEUS 6, WebSphere 6.1, Tomcat 8.5가 함께 기재돼 있습니다[S36][S38][S40]. 이 스택을 버전 그대로 컨테이너 이미지로 만들면 전환 사업의 범위가 줄어드는 것처럼 보입니다. 하지만 버전을 유지한 컨테이너 전환은 다음 세 가지 이유로 성립하지 않습니다.

- 전환 대상 대부분은 운영체제, Java, WAS, 프레임워크 네 계층 가운데 둘 이상에서 제조사와 커뮤니티의 지원이 이미 끝난 상태입니다.
- JVM이 컨테이너의 자원 한도를 기본으로 인식하는 것은 JDK 10부터이고, JDK 8 계열에서는 8u191부터입니다[S07].
- 상용 WAS에 종속된 코드와 구성은 목표 환경에서 빌드와 실행을 막기 때문에 선택 항목이 될 수 없는 필수 수정 대상입니다.

따라서 현대화를 뒤로 미루어도 그 비용은 없어지지 않고 다음 사업의 범위로 남습니다. 전환 범위에 버전 상향을 포함할지 판단하려면 계층별 지원 종료 일정, JDK 버전별 컨테이너 인식 기능, Java 상향 경로별 수정 항목, 순수 Java와 네이티브 코드의 전환 난이도 차이를 함께 확인해야 합니다. 이 네 가지 근거는 사전 진단에서 Java 업데이트 번호, WAS 종속 항목, 네이티브 라이브러리 유무를 수집하는 이유이기도 합니다.

5.1 운영체제에서 프레임워크까지 네 계층의 지원 종료 일정

전환 대상에서 반복해 발견되는 구버전 스택은 운영체제, Java, WAS, 프레임워크 네 계층으로 나누어 볼 수 있습니다. 지원 종료는 보안 패치와 결함 수정이 더 제공되지 않는다는 뜻이므로, 구버전을 그대로 옮기면 전환 직후부터 같은 보안 위험을 새 환경에서 다시 부담합니다. 공개된 상세설계 제안요청서의 대상 시스템을 제조사 수명주기 안내와 대조하면 대부분이 네 계층 가운데 둘 이상에서 지원이 끝나 있습니다. 2024년 상세설계 4차 사업의 한 대상은 OpenJDK 1.6, 전자정부 표준프레임워크 1.2, JEUS 6 조합으로 기재돼 있습니다[S40].

5.1.1 CentOS 7·AIX 7.1로 대표되는 운영체제 지원 종료 시점

CentOS의 지원 종료일

CentOS Linux 8은 2021년 12월 31일에 지원이 끝났습니다[S10]. CentOS 7도 2024년 6월 30일에 지원이 끝난 것으로 집계됩니다[S18][S45]. 전환 대상에서 자주 확인되는 두 버전이 모두 지원 기간을 지난 상태입니다. 상용 리눅스 배포판의 지원 종료일은 공급자의 수명주기 공지로 확인합니다.

AIX 7.1과 Unix 계열의 지원 종료일

Unix 계열도 상황이 같습니다. IBM의 수명주기 안내에 따르면 AIX 7.1의 마지막 기술 수준인 TL5(Technology Level 5)는 2023년 4월 30일에 지원이 끝났습니다[S11]. 2026년 상세설계 사업의 대상에도 HP-UX 11.31과 AIX 7.1·5.3에서 운영 중인 시스템이 들어 있으므로, 지원이 끝난 Unix는 현재 발주되는

사업에서 그대로 다루어야 하는 대상입니다[S36]. 아래 표는 운영체제별 종료를 확인 수준과 함께 정리한 것입니다.

운영체제	지원 종료 일정	확인 수준
CentOS Linux 8	2021-12-31 종료	프로젝트 공식 공지[S10]
CentOS 7	2024-06-30 종료	2차 확인(집계 사이트와 NIA 가이드 기재)[S18][S45]
AIX 7.1 (TL5)	2023-04-30 종료	제조사 공식 안내[S11]
HP-UX 11i v3	표준 지원 2025-12-31 종료	2차 확인(NIA 가이드 기재)[S45]

표에서 「2차 확인」으로 적은 값의 출처는 집계 사이트와 NIA 가이드이며 제조사 페이지에서 직접 확인한 값과 구분해야 합니다. 이 값을 발주 문서에 인용할 때는 제조사 공식 페이지와 대조하는 절차가 필요합니다.

전환 방식별 운영체제 문제의 처리

운영체제 지원 종료 문제는 전환 방식에 따라 처리 결과가 다릅니다. 컨테이너로 전환하면 목표 환경의 호스트 운영체제와 컨테이너 기반 이미지가 지원 중인 버전으로 바뀌므로 이 문제를 전환과 함께 해소할 수 있습니다. 반면 가상 머신(VM, Virtual Machine)으로 이전하면 게스트 운영체제가 그대로 유지되므로 지원이 끝난 운영체제도 새 환경에 그대로 남습니다. 이 차이는 컨테이너 전환을 기본으로 정하고 가상화 전환을 사유가 확인된 예외로 한정하는 근거 가운데 하나입니다(6장).

5.1.2 Tomcat 8.5·Spring Boot 2.7의 지원 종료 시점과 Java 17 상향 근거

OpenJDK 배포판의 지원 기간

Java의 지원 기간은 배포판마다 다르므로 어느 배포판을 기준으로 삼았는지를 먼저 밝혀야 합니다. OpenJDK 8.11의 지원 종료 시점은 배포판 공급자마다 다르므로 사용 중인 배포판의 수명주기 공지를 확인해야 합니다. 목표 Java를 8이나 11로 정하면 전환을 마친 뒤 프레임워크를 상향할 때 Java를 다시 올려야 합니다. Spring Boot 3.0이 Java 17을 요구하기 때문입니다[S13]. 목표 Java를 17 이상으로 정해야 프레임워크 상향과 운영기간을 함께 확보할 수 있습니다.

Tomcat과 Spring의 지원 종료

WAS와 프레임워크 계층도 같은 방향입니다. Apache Tomcat 8.5는 2024년 3월 31일에 지원이 끝났고, Tomcat 9는 Java EE 8을 지원하는 마지막 메이저 버전입니다[S15]. Spring Framework 5.3은 2024년 8월에 오픈소스 지원이 끝났습니다[S14]. 상용 WAS 가운데 IBM WebSphere Application Server 7과 8은 2018년 4월 30일에 지원이 끝났는데, 2026년 상세설계 대상에는 그보다 앞선 WebSphere 6.1이 기재돼 있습니다[S17][S36]. 다만 JEUS 6·7·8은 공개된 지원 종료 공지를 확인하지 못했으므로 제조사 확인이 필요한 항목으로 남겨 둡니다.

계층	대상	지원 종료 일정	기준과 확인 수준
WAS	Apache Tomcat 8.5	2024-03-31 종료	Apache 공식 안내[S15]
WAS	IBM WebSphere 7·8	2018-04-30 종료	제조사 공식 안내[S17]

계층	대상	지원 종료 일정	기준과 확인 수준
프레임워크	Spring Framework 5.3	오픈소스 지원 2024-08 종료	Spring 공식 안내[S14]
프레임워크	Spring Boot 2.7	오픈소스 지원 2023-06-30 종료	2차 확인(집계 사이트) [S18]

운영체제·Java·WAS·프레임워크 네 계층의 지원 종료 시점

막대의 오른쪽 끝이 지원 종료 시점입니다. 지원 시작 시점은 표시하지 않았습니다.

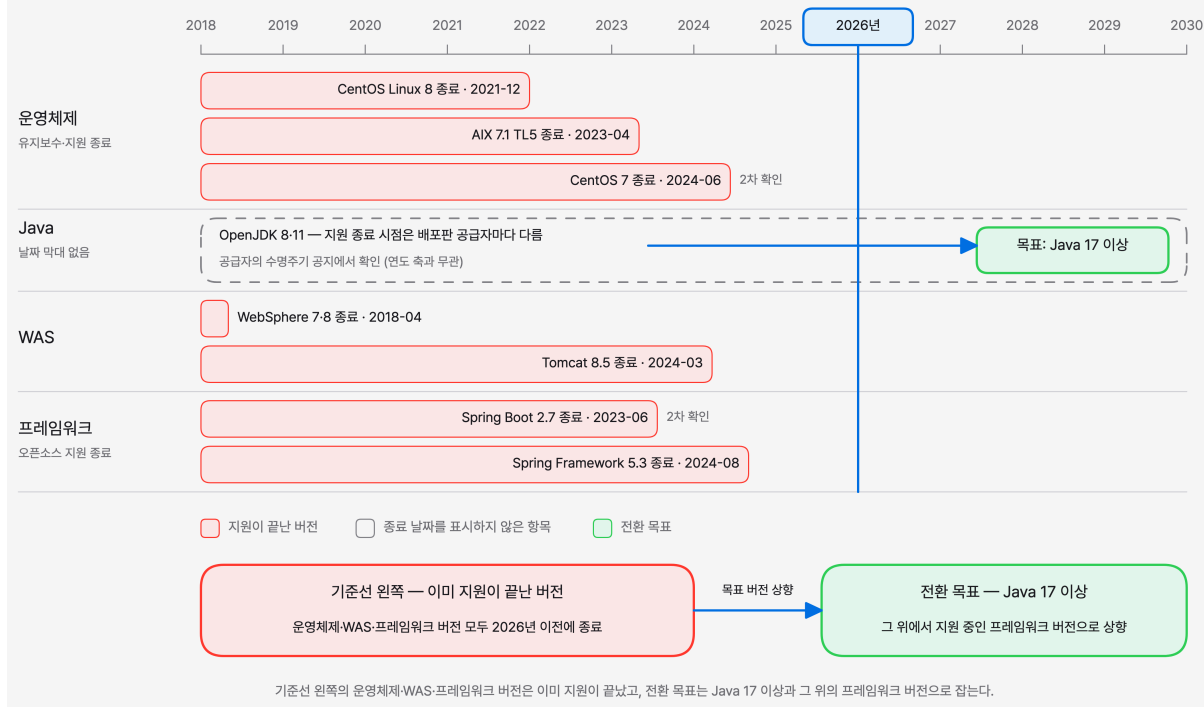


그림. 운영체제·WAS·프레임워크의 지원 종료 시점과 Java 목표 버전 (CentOS 7과 Spring Boot 2.7의 종료 시점은 집계 사이트로 확인한 2차 확인 값)

목표 버전 선택과 운영 기간

목표 버전을 정할 때 비교할 값은 현재 버전의 종료일보다 목표 버전의 잔여 지원 기간입니다. 예를 들어 Java 8에서 11로만 상향하면 수정 범위는 작지만 Java 11은 Spring Boot 3.0이 요구하는 Java 17에 미치지 못하므로 운영 단계에서 다시 상향 사업을 발주해야 합니다. 사전 진단이 목표 Java를 17 이상으로 놓았을 때의 수정 범위를 기준으로 공수를 산정하는 것도 이 때문입니다.

5.1.3 전자정부 표준프레임워크 4.x와 Java 17 상향이 묶이는 구조

프레임워크 상향과 Java 상향의 결합

공공 시스템에서는 프레임워크 상향과 Java 17 상향을 분리해 발주할 수 없습니다. 전자정부 표준프레임워크 4.3.0(2024년 5월)은 Spring Framework 5.3.37과 Spring Boot 2.7.18을 기반으로 하고, 5.0.0은 Java 17 이상과 Spring Framework 6.2를 기반으로 합니다[S16]. 4.x는 오픈소스 지원이 끝난 Spring 5.3 계열을 사용하므로, 지원 중인 프레임워크로 상향하려면 5.0.0으로 가야 하고 5.0.0은 Java 17을 전제로 합니다.

Spring Boot 3.0도 Java 17과 Spring Framework 6을 요구하므로 표준프레임워크를 쓰지 않는 시스템에서도 결론은 같습니다[S13].

표준프레임워크 버전	기반 Spring·Java 버전	기반 Spring의 지원 상태
4.3.0 (2024-05)	Spring Framework 5.3.37, Spring Boot 2.7.18	Spring Framework 5.3은 2024-08에 오픈소스 지원 종료[S14]
5.0.0	Spring Framework 6.2, Java 17 이상	지원 중인 Spring Framework 6 계열

공개 발주 문서의 표준프레임워크 버전

공개된 상세설계 제안요청서의 대상 시스템은 4.x보다도 낮은 버전을 쓰고 있습니다. 2026년 사업 대상에는 표준프레임워크 3.0.5가, 2024년 3차와 4차 사업 대상에는 각각 3.9와 1.2가 기재돼 있습니다[S36][S39][S40]. 이 시스템들은 프레임워크를 두 세대 이상 상향해야 하고, 그 과정에서 Java 17 상향을 함께 수행해야 합니다.

발주 문서의 목표 버전 기준

이 결합 관계 때문에 목표 버전은 발주 단계에서 정해야 하는 의사결정 사항입니다. 발주 문서에 목표 프레임워크와 Java 버전을 정하지 않으면 사업자마다 다른 목표 버전을 가정해 견적을 내고, 그 결과 제안 금액을 같은 기준으로 비교할 수 없습니다. 한 사업자는 표준프레임워크 4.3.0 유지를, 다른 사업자는 5.0.0 상향을 전제로 공수를 산정할 수 있습니다. 사전 진단은 목표 버전을 먼저 정한 뒤 그 버전을 기준으로 수정 범위를 계량합니다.

5.2 JVM이 컨테이너 자원 한도를 인식하기 시작한 JDK 버전

컨테이너는 리눅스 커널의 cgroup(control group, 프로세스 묶음에 CPU와 메모리 한도를 적용하는 기능)으로 자원을 제한합니다. JVM이 이 한도를 읽지 못하면 호스트 전체의 메모리와 CPU를 자기 자원으로 계산하고, 그 결과 한도를 넘는 메모리를 사용해 커널이 컨테이너를 강제 종료합니다. JVM에 컨테이너 인식 기능이 반영된 버전은 OpenJDK 이슈 기록으로 확인할 수 있으며, 기준은 JDK 10과 8u191입니다[S07]. Java 1.6과 1.7에는 컨테이너 인식 기능이 전혀 없으므로, 이 버전의 애플리케이션은 컨테이너 전환과 Java 상향을 분리할 수 없습니다.

5.2.1 JDK 10과 8u191부터 기본 활성화된 컨테이너 지원

컨테이너 지원의 반영 버전

컨테이너를 감지해 자원 설정에 반영하는 기능(UseContainerSupport)은 JDK 10에서 기본으로 활성화됐고, JDK 8 계열에는 8u191에 반영됐습니다[S07]. 그 전 단계로 cgroup 메모리 한도를 힙 계산에 쓰는 실험 옵션(UseCGroupMemoryLimitForHeap)이 JDK 9와 8u131에 들어갔습니다[S08]. 이 실험 옵션은 실행 인자로 명시해야 동작하므로 기본 설정으로 실행하는 애플리케이션에는 적용되지 않습니다.

기능	반영 버전	JDK 8 계열	근거
cgroup 메모리 한도 실험 옵션	JDK 9	8u131	JDK-8170888[S08]

기능	반영 버전	JDK 8 계열	근거
(UseCGroupMemoryLimitForHeap)			
컨테이너 지원 기본 활성화 (UseContainerSupport)	JDK 10	8u191	JDK-8146115[S07]
cgroup v2 인식	JDK 15, 11.0.16	8u372	JDK-8230305[S09]

「Java 11부터 인식한다」는 통념의 정정

현장에서는 「Java 11부터 컨테이너를 인식한다」는 설명이 자주 쓰이지만 이 설명은 정확하지 않습니다. 기본 활성화의 기준은 JDK 10과 8u191이므로, Java 8을 쓰더라도 8u191 이상이면 cgroup v1 환경에서 한도를 인식합니다[S07]. Java 8이라는 메이저 버전만으로는 인식 여부를 알 수 없고, 8u131보다 낮은 업데이트는 어떤 옵션으로도 한도를 읽지 못합니다. 그래서 사전 진단에서는 Java 메이저 버전과 함께 업데이트 번호를 수집합니다.

5.2.2 cgroup v2 인식에 필요한 JDK 15·11.0.16·8u372

cgroup v2의 반영 버전

cgroup에는 v1과 v2 두 가지 구조가 있고, JVM은 두 구조를 서로 다른 코드로 읽습니다. cgroup v2 인식은 JDK 15에 들어갔고 이후 11.0.16과 8u372에도 들어갔습니다[S09]. 최신 리눅스 배포판과 그 위에 구성된 Kubernetes 노드는 cgroup v2를 기본으로 사용합니다. 따라서 Java 11을 쓰고 있더라도 업데이트 번호가 11.0.15 이하이면 목표 환경에서 한도를 읽지 못할 수 있습니다.

이슈 기록의 사실과 추론의 구분

이슈 기록에 적힌 사실은 cgroup v2 인식 기능이 JDK 15, 11.0.16, 8u372에 들어갔다는 것까지입니다. 「11.0.15 이하와 8u362 이하는 cgroup v2 호스트에서 한도를 읽지 못한다」는 내용은 이 기록에서 끌어낸 추론이며, 배포판이 자체적으로 수정 사항을 앞선 버전에 반영했을 가능성은 배포판별로 확인해야 합니다. 아래 표는 이 구분을 전제로 호스트의 cgroup 버전과 JDK 업데이트 버전 조합별 인식 여부를 정리한 것입니다.

JDK 버전	cgroup v1 호스트	cgroup v2 호스트
Java 1.6~1.7, 8u131 미만	인식하지 못함	인식하지 못함
8u131~8u190, JDK 9	실험 옵션을 지정한 경우에만 메모리 한도 반영	인식하지 못하는 것으로 추론
8u191부터 8u362까지, JDK 10부터 14까지, 11.0.15 이하	기본으로 인식	인식하지 못하는 것으로 추론
8u372 이상, 11.0.16 이상, JDK 15 이상	기본으로 인식	기본으로 인식

업데이트 번호 수집의 필요성

표의 셋째 줄은 Java 메이저 버전만 확인하는 조사에서 누락되는 조합입니다. Java 8이나 11을 사용한다는 사실만 확인하고 전환하면, 기존 cgroup v1 환경의 시험에서는 정상이던 애플리케이션이 cgroup v2를 쓰는 운영 환경에서 메모리 한도를 넘을 수 있습니다. 사전 진단은 1.8.0_181 이나 11.0.9 같은 전체 버전 문자열을 수집하고, 목표 환경의 cgroup 버전과 대조해 상향 대상 여부를 판정합니다.

5.2.3 힙을 호스트 메모리 기준으로 계산해 발생하는 OOMKilled 사례

호스트 메모리 기준의 힙 계산

컨테이너를 인식하지 못하는 JVM은 최대 힙 크기를 컨테이너 한도 대신 호스트의 물리 메모리 기준으로 계산합니다. 이런 JVM은 최대 힙의 기본값을 물리 메모리의 4분의 1 수준으로 잡습니다. 예를 들어 메모리 8GB 호스트에서 한도 600MB로 컨테이너를 실행하면 JVM은 최대 힙을 약 2GB로 계산하고, 애플리케이션이 메모리를 600MB 넘게 쓰는 순간 커널이 컨테이너를 강제 종료합니다. Kubernetes는 이 상태를 OOMKilled(Out Of Memory Killed, 메모리 한도 초과로 인한 강제 종료)로 기록합니다.

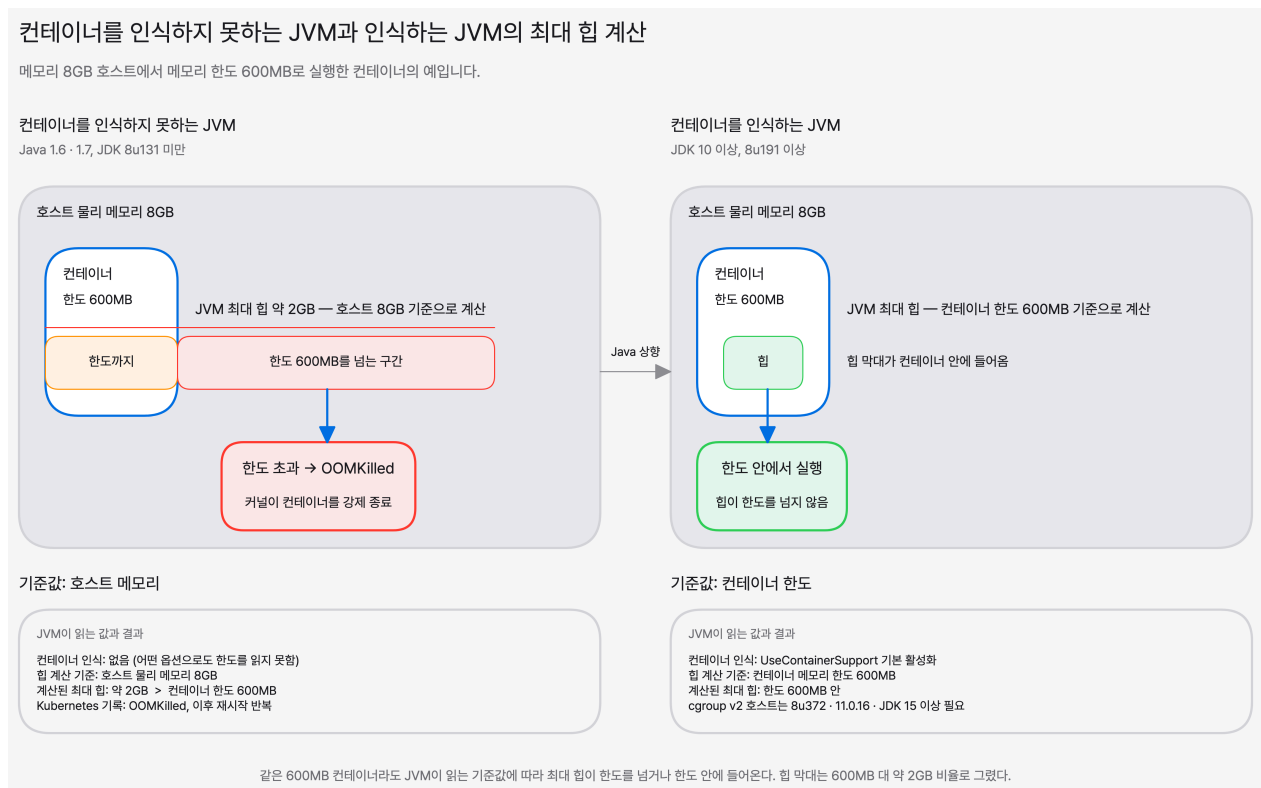


그림. 컨테이너를 인식하지 못하는 JVM과 인식하는 JVM의 최대 힙 계산 비교

전환 직후 장애로 이어지는 과정

이 장애는 전환 직후에는 드러나지 않다가 부하가 늘어나는 시점에 발생합니다. 힙 사용량이 한도보다 작은 동안에는 정상으로 동작하므로 기능 시험만으로는 발견하기 어렵습니다. 강제 종료된 컨테이너는 Kubernetes가 다시 시작하므로, 원인을 모르는 운영자가 확인할 수 있는 것은 반복되는 재시작뿐입니다. 구버전 Java를 그대로 컨테이너로 옮긴 시스템은 운영 단계에서 장애 대응 비용을 추가로 부담합니다.

공공 전환 대상의 Java 버전 분포

이 문제는 실제 발주 대상에 해당합니다. 2024년 상세설계 2차 사업의 대상 시스템에는 JDK 1.6·1.7·1.8이 혼재하고, 4차 사업 대상에는 OpenJDK 1.6이, 1차 사업 대상에는 JDK 1.7이 기재돼 있습니다[S37][S38][S40]. Java 1.6과 1.7은 실험 옵션조차 없는 버전이므로 힙 크기를 실행 인자로 직접 고정하는 임시 조치만 가능합니다. 이 버전의 애플리케이션은 Java 상향을 컨테이너 전환의 선행 조건으로 계획해야 합니다.

5.3 상용 WAS 종속 코드가 필수 수정 대상이 되는 근거

상용 WAS 교체는 라이선스 비용의 문제로 논의되는 경우가 많지만, 전환 사업의 범위를 결정하는 것은 코드 수정 범위입니다. 상용 WAS의 전용 기능에 종속된 코드와 구성은 목표 환경에 해당 기능이 없으므로 수정하지 않으면 빌드되거나 실행되지 않습니다. CNAMT 문서는 이런 항목을 Mandatory(필수)로 분류하고, 대표 예로 대상 환경이 지원하지 않는 독점 API를 듭니다. 이 수정 범위는 정적 분석으로 사전에 계량할 수 있으므로 본사업 발주 전에 확정할 수 있습니다.

5.3.1 전용 배포 서술자·JNDI·독점 패키지 import의 표준 방식 전환

Mandatory로 분류되는 종속 항목

상용 WAS 종속 항목은 크게 세 가지입니다. 첫째는 제품 전용 배포 서술자(deployment descriptor, 애플리케이션의 배포 설정을 적은 파일)이고, 둘째는 전용 JNDI(Java Naming and Directory Interface, 이름으로 자원을 찾는 표준 인터페이스) 프로토콜이며, 셋째는 제품 독점 패키지를 import한 소스 코드입니다. Apache Tomcat 같은 목표 WAS는 이 전용 형식과 패키지를 지원하지 않으므로 세 항목 모두 Mandatory 정의에 해당합니다. 수정하지 않은 채 컨테이너 이미지를 만들면 기동 단계에서 오류가 발생합니다.

상용 WAS 종속 항목	표준 대체 방식	수정 위치
전용 배포 서술자	표준 서술자와 목표 WAS의 설정 형식으로 재작성	설정 파일
전용 JNDI 프로토콜	표준 JNDI 조회 방식과 목표 WAS의 자원 정의로 변경	설정 파일, 소스 코드
독점 패키지 import	표준 API 또는 공개 라이브러리로 교체	소스 코드
웹서버 연동 프로토콜	목표 웹서버와 WAS 조합이 지원하는 연동 방식으로 재설정	웹서버·WAS 설정
클라이언트 IP 헤더	목표 환경의 프록시가 전달하는 헤더 기준으로 변경	설정 파일, 소스 코드

전환 항목의 규모와 추가 공수

EJB와 JMS를 쓰지 않는 애플리케이션이라면 전환 항목의 수는 많지 않습니다. 기관명 비공개 Unix 전환 계획의 사례에서 상용 WAS를 오픈소스 계열 WAS로 바꾸는 작업은 배포 서술자, JNDI, 웹서버 연동 프로토콜, 클라이언트 IP 헤더를 포함한 여섯 항목 수준이었습니다. 다만 JMS를 사용하면 메시지 처리 구성을 새로 설계해야 하므로 5~10일을 추가로 산정합니다. 모듈마다 Spring 버전의 편차가 크면 라이브러리 호환 검증을 별도 작업으로 산정합니다.

공공 가이드의 공수 예시와 판정

정부 측 기준에도 같은 항목이 있습니다. NIA의 클라우드 네이티브 구축·운영 가이드(2025년 5월)에 실린 설계 기준 예시는 국산·공개 WEB·WAS 적용을 난이도 중, 서버당 9 M/D로 적습니다. 같은 가이드는 「애플리케이션의 WAS 라이브러리 종속 수정」을 전환불가 판정 예시에 포함합니다[S45]. 반면 정적 분석은 이 항목을 건수와 점수로 계량하므로, 사전 진단을 거치면 전환불가로 분류될 대상을 수정 범위가 정해진 컨테이너 전환 대상으로 다시 판정할 수 있습니다.

5.3.2 javax에서 jakarta로 바뀐 네임스페이스와 Spring Boot 3의 Java 17 전제

Jakarta EE 9의 네임스페이스 변경

네임스페이스는 Java 패키지 이름의 앞부분으로, 소스 코드의 import 문에 그대로 적히는 값입니다. Jakarta EE 9는 2020년 12월에 이 네임스페이스를 javax 에서 jakarta 로 일괄 변경했습니다[S12]. Apache Tomcat은 10부터 jakarta 네임스페이스를 사용하므로 Tomcat 9까지 동작하던 애플리케이션은 Tomcat 10 이상에서 그대로 실행되지 않습니다[S15]. Spring Boot 3.0도 Java 17과 Spring Framework 6을 요구하며 같은 네임스페이스 변경을 전제로 합니다[S13].

목표 버전에 따른 작업 범위의 변화

네임스페이스 변경은 특정 모듈에 한정되지 않으며 서블릿, 영속성, 검증 기능을 쓰는 모든 소스 파일이 수정 대상입니다. 그래서 목표를 Java 17과 Spring Boot 3으로 정하면 4장에서 설명한 「생략 가능」 점수가 작업 대상으로 바뀝니다. 공공기관 D의 사전 진단에서는 Story Point 937점 가운데 920점이 생략 가능 점수였는데, 이 분류는 javax 에서 jakarta 로의 변경과 JDK 11에서 제거된 모듈을 전환 범위에서 제외한다는 전제에 따른 것입니다. 즉 목표 Java 버전을 어디로 정하는지가 공수 산정 결과를 좌우합니다.

Java 상향 경로별 수정 항목

Java 상향은 경로마다 수정 항목의 성격이 다릅니다. 아래 표는 세 경로의 대표 수정 항목을 정리한 것입니다.

상향 경로	대표 수정 항목	부담 수준
Java 8 → 11	Java EE-CORBA 모듈(JAXB, JAX-WS 등) 제거에 따른 의존성 추가, 내부 API 접근 제한 대응	의존성과 빌드 설정 수정
Java 11 → 17	강한 캡슐화로 인한 리플렉션 접근 예외 수정	소스 수정, 실행 옵션 우회는 임시 조치
Java 17 → 21	가상 스레드 등 신규 기능 도입 검토	호환성 부담이 작음

표에서 부담이 큰 경로는 8에서 11, 11에서 17까지의 두 단계이며 17에서 21로의 상향은 상대적으로 부담이 작습니다. 목표를 17로 정한 사업은 이후 21로 상향할 때 큰 수정 없이 진행할 수 있습니다.

5.3.3 세션 복제와 배포 관리가 WAS에서 Kubernetes로 넘어가는 책임 이동

WAS와 Kubernetes의 실행 모델 차이

WAS는 JVM 하나에 여러 애플리케이션을 배포하고 WAS가 그 수명주기를 관리하는 모델입니다. 반면 Kubernetes는 컨테이너 하나에 프로세스 하나가 실행되고, 컨테이너의 수를 늘리거나 교체하는 방식으로 확장과 배포를 처리합니다. 이 차이 때문에 WAS가 맡던 세션 복제, 배포 관리, 자원 설정 기능은 컨테이너 전환 후 플랫폼의 책임으로 이동합니다. WAS 클러스터가 인스턴스 사이에 세션을 복제하던 구성은 세션을 외부 저장소에 두는 구성으로 바뀝니다.

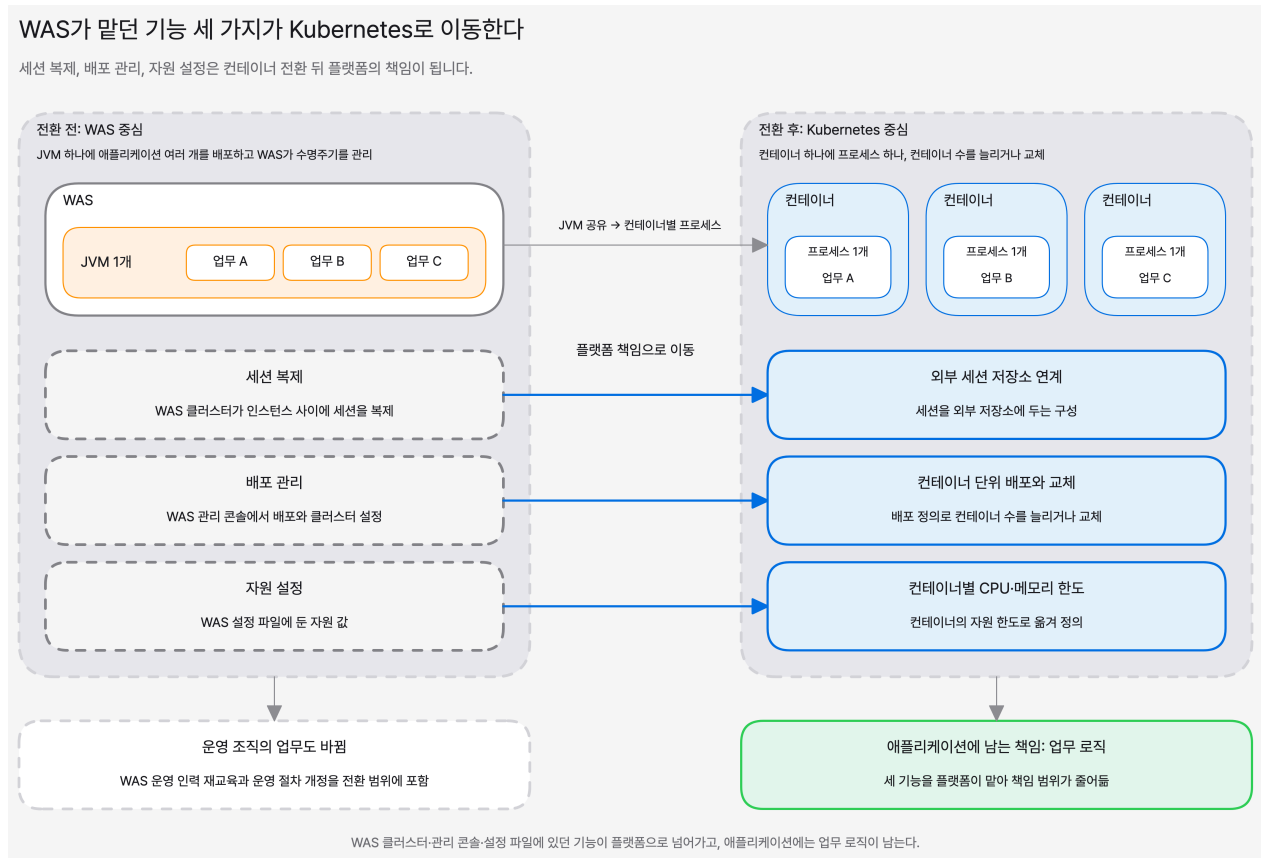


그림. WAS가 맡던 세션 복제·배포 관리·자원 설정이 Kubernetes로 이동하는 구조

운영 조직의 역할 변화

책임 이동은 기술 구성과 함께 운영 조직의 업무를 바꿉니다. WAS 관리 콘솔에서 수행하던 배포와 클러스터 설정 업무는 Kubernetes의 배포 정의를 관리하는 업무로 바뀝니다. 5.2절에서 설명한 메모리 한도처럼 WAS 설정 파일에 두던 자원 값도 컨테이너의 자원 한도로 옮겨 정의합니다. 따라서 WAS 운영 인력의 재교육과 운영 절차 개정을 전환 범위에 포함해야 합니다.

전환 범위에 포함할 운영 항목

전환 범위를 애플리케이션 수정으로만 정하면 이 운영 항목이 범위에서 빠집니다. 그 결과 전환을 마친 뒤에도 운영 조직이 기존 WAS 방식으로 배포와 장애 대응을 계속합니다. 사전 진단은 세션 사용 방식, 배포 절차, 자원 설정 위치를 체크리스트로 조사해 본사업 범위에 반영할 항목을 정리합니다(3장).

5.4 순수 Java와 플랫폼 종속 네이티브 코드의 Unix→Linux 전환 난이도 차이

Unix에서 리눅스로 옮기는 U2L(Unix to Linux) 전환에서 난이도를 가르는 기준은 플랫폼 의존도입니다. 순수 Java 애플리케이션은 JVM이 운영체제 차이를 흡수하므로 이식 난이도가 낮고, 플랫폼에 종속된 네이티브 코

드는 소스 수정과 재컴파일이 필요하므로 난이도가 높습니다. Unix에서 운영 중인 Java 애플리케이션은 가상화 예외 대상에 해당하지 않으며 리눅스 컨테이너에서의 동작 검증 대상입니다. 반면 네이티브 코드의 유무는 전환 범위와 예외 판정에 직접 영향을 주므로 사전 진단의 확인 항목입니다.

5.4.1 JVM 명세가 보장하는 순수 Java 애플리케이션의 이식성

JVM 명세와 포팅 가이드의 근거

순수 Java 애플리케이션의 이식성은 JVM 명세가 보장합니다. Java Virtual Machine Specification은 JVM을 하드웨어와 운영체제로부터의 독립성을 담당하는 요소로 정의합니다[S20]. JVM이 운영체제의 차이를 흡수하므로 순수 Java 애플리케이션은 Unix에서 리눅스로 옮길 때 이식 난이도가 낮은 대상에 속합니다.

이식 난이도의 구분 기준

이식 난이도를 나누는 기준은 언어의 종류보다 이식성과 의존성입니다. 소스 코드가 없는 애플리케이션은 다시 빌드할 수 없으므로 x86용 제품의 제공 여부가 선택지를 결정합니다. 아래 표는 이 기준을 전환 대상 유형별로 정리한 것이며, 인용한 포팅 가이드는 2005년 문서이므로 원칙의 근거로만 사용합니다.

대상 유형	이식 난이도	근거
순수 Java 애플리케이션	낮음	JVM이 하드웨어·운영체제 독립성을 담당(JVM 명세)[S20]
JNI·벤더 JDK 고유 클래스를 쓰는 Java 애플리케이션	확인 필요	5.4.2항의 확인 항목에 따라 판정
플랫폼 종속 C 코드	높음	엔디언·시스템 콜 차이로 소스 수정 필요(IBM, 2005)[S19]
소스 코드가 없는 애플리케이션	x86용 제품 유무에 따라 결정	다시 빌드할 수 없어 6.3.2항의 CPU 아키텍처 단서 적용

Unix 위 Java 애플리케이션의 전환 방식

이 근거에 따르면 AIX나 HP-UX에서 운영 중인 Java 애플리케이션을 전환할 때의 핵심 작업은 리눅스에서의 정상 동작 검증이며 재개발을 전제로 하지 않습니다. 예를 들어 대학 E의 사전 진단에서는 AIX에서 운영 중인 Java 애플리케이션도 리눅스 환경의 애플리케이션과 같은 방법으로 분석하고 판정했습니다. 그래서 「Unix에서 운영 중」이라는 사실만으로는 가상화 전환의 사유가 되지 않습니다. 예외 사유의 판정 기준은 6장에서 다룹니다.

5.4.2 JNI·벤더 JDK 고유 클래스·기본 문자셋에서 발생하는 Java의 예외 지점

JNI 네이티브 라이브러리

「순수 Java」라는 한정이 필요한 첫째 이유는 JNI(Java Native Interface, Java에서 C 등으로 작성한 네이티브 코드를 호출하는 인터페이스)입니다. JNI 명세가 보장하는 호환 범위는 하나의 플랫폼 안입니다[S21]. JNI로 호출하는 네이티브 라이브러리는 플랫폼이 바뀌면 C 코드와 같은 재컴파일과 수정 문제를 겪습니다. 사전 진단 결과에서 JNI 사용이 「확인 필요」로 표시되는 것은 이 때문입니다.

벤더 JDK 고유 클래스와 기본 문자셋

둘째는 벤더 JDK의 고유 클래스입니다. IBM은 자사 JDK에서 IBM Semeru Runtimes로 옮길 때 `com.ibm.net`, `com.ibm.nio`, `com.ibm.lang` 패키지가 제공되지 않고 보안 구현이 달라진다고 안내합니다[S23]. AIX의 IBM Java에서 OpenJDK로 전환하는 경우가 여기에 해당합니다. 셋째는 기본 문자셋입니다. JDK 17 이하는 기본 문자셋이 운영체제와 로케일(언어·지역 설정)을 따르므로, 기본값이 다른 환경으로 옮기면 텍스트가 오류 없이 손상될 수 있습니다[S22]. EUC-KR이나 MS949를 기본으로 쓰던 환경에서 UTF-8 환경으로 옮길 때 이 위험이 발생합니다.

확인 항목	전환 후 증상	확인 방법
JNI 네이티브 라이브러리	라이브러리를 불러오지 못해 기동 또는 해당 기능 호출이 실패	배포본의 네이티브 라이브러리 파일 목록과 네이티브 메서드 선언 조사
벤더 JDK 고유 클래스	클래스를 찾지 못해 빌드 또는 실행이 실패	소스와 라이브러리에서 <code>com.ibm.*</code> 등 벤더 패키지 import 검색
기본 문자셋	오류 메시지 없이 한글 텍스트가 손상	현재 운영체제 로케일과 JVM 기본 문자셋 설정, 파일·연계 데이터의 인코딩 확인

세 항목의 진단 시점

세 항목은 모두 본사업 착수 전에 확인할 수 있습니다. 네이티브 라이브러리와 벤더 패키지는 배포본과 소스의 검색으로 유무를 판정할 수 있고, 문자셋은 운영 서버의 설정값 조회로 확인할 수 있습니다. 반면 이 확인을 본사업으로 미루면 통합 시험 단계에서 문자 손상이나 기동 실패로 드러나 일정에 영향을 줍니다. 사전 진단이 세 항목을 Java 애플리케이션의 필수 확인 항목으로 두는 이유입니다.

5.4.3 C 언어 재컴파일 9 M/D로 본 네이티브 코드 전환 공수

NIA 설계기준 예시의 난이도와 공수

네이티브 코드 전환의 공수를 제시한 공개 수치는 NIA 클라우드 네이티브 구축·운영 가이드(2025년 5월) 부록의 설계기준 예시에 있습니다. 이 예시는 Unix→Linux 운영체제 전환을 난이도 상으로, C 언어 재컴파일을 난이도 상·9 M/D로, JDK 버전 변경에 따른 프레임워크 버전업을 난이도 상·17 M/D로 적습니다[S45]. 17 M/D는 월 20일 기준으로 0.85 M/M에 해당합니다. 다만 이 표는 IaaS 전환용 예시이며 컨테이너 전환용 대응표는 가이드에 없으므로, 수치는 항목 간 상대 비교의 참고값으로 사용해야 합니다.

구분	작업	난이도	서버당 M/D
OS	Unix → Linux	상	2
OS	Linux → Linux	중	1
WEB·WAS	국산·공개 WEB·WAS 적용	중	9
AP	C 언어 재컴파일(Unix → Linux)	상	9
AP	런타임 버전 업그레이드 호환 확인	상	17

구분	작업	난이도	서버당 M/D
AP	프레임워크 버전업(JDK 버전 변경에 따른 코어 라이브러리·소스 수정)	상	17

위 표는 가이드의 예시표에서 5장과 관련된 항목만 발췌한 것입니다[S45].

플랫폼 종속 C 코드의 수정 원인

플랫폼에 종속된 C 코드는 재컴파일만으로 끝나지 않고 소스 수정이 따릅니다. IBM이 2005년에 발행한 Solaris→Linux 포팅 가이드는 엔디언(바이트 저장 순서), 구조체 패딩, 시스템 자료형 크기, 시그널, 시스템 콜, 스레드 라이브러리, 컴파일러의 차이를 수정 원인으로 듭니다[S19]. 이 가운데 엔디언 차이는 네트워크 통신과 디스크에 저장된 데이터의 해석에 영향을 줍니다. 국내 증권사의 코어시스템 U2L 전환 보도에서도 AP(애플리케이션) 서버가 플랫폼에 종속적인 C 언어로 개발돼 소스 프로그램의 상당 부분을 수정해야 했다는 설명이 있습니다[S24].

배수 표현을 쓰지 않는 근거

언어별 포팅 공수를 직접 비교한 공개 수치는 위 예시표 외에 확인되지 않습니다. 그래서 「네이티브 코드는 Java보다 몇 배의 공수가 든다」 같은 배수 표현은 근거가 없으며, 사전 진단 결과에도 쓰지 않습니다. 2026년 상세설계 사업의 대상처럼 언어 구성에 C가 포함된 Unix 시스템에서는 네이티브 코드의 분량과 플랫폼 의존 항목을 개별로 조사해 공수를 산정합니다[S36]. Java 부분은 정적 분석 점수로, 네이티브 코드 부분은 항목별 조사로 계량하고 두 결과를 합쳐 전환 범위를 정합니다.

6장. 전환 원칙: 컨테이너 전환이 기본, 가상화 전환은 예외

5장에서 본 대로 구버전 스택을 그대로 둔 전환은 현대화 비용을 다음 사업으로 넘깁니다. 대상 시스템을 컨테이너로 옮기든지 가상 머신(VM)에 그대로 두는지는 전환 사업의 범위와 비용을 가장 크게 좌우합니다. 컨테이너로 옮기려면 애플리케이션을 수정해야 하고, VM으로 옮기면 수정 없이 인프라만 바뀝니다. 일정이 촉박한 사업일수록 VM 이전이 선택되기 쉽고, 그 결과 지원이 끝난 운영체제와 미들웨어가 새 인프라에 그대로 남습니다. 그래서 이 백서는 컨테이너 전환을 기본으로 두고 가상화 전환을 사유가 확인된 예외로만 허용하며, 그 원칙과 대상별 판정 기준, 예외 사유와 기록 항목을 아래에 제시합니다. 의사결정권자는 6.4절의 예외 사유표를 조직의 판정 기준으로 채택할지 정하게 됩니다.

6.1 모든 대상에 컨테이너 전환을 먼저 검토하는 세 가지 원칙

컨테이너 우선 원칙은 전환 대상마다 컨테이너 전환 가능성을 먼저 확인하고, 가상화 전환은 정해진 사유가 확인된 대상에만 허용하는 판정 규칙입니다. NIA가 2021년에 펴낸 클라우드 네이티브 정보시스템 구축 발주자 안내서는 서문에서 기술 인프라 위주의 클라우드 도입보다 클라우드 네이티브 정보시스템 구축이 필요하다고 밝혔습니다[S43]. 인프라만 바꾸는 이전으로는 정부가 요구하는 수준에 도달하지 못한다는 뜻이며, 컨테이너 우선 원칙은 그 요구를 대상별 판정 절차로 옮긴 것입니다. 원칙이 정해지면 사전 진단이 답해야 할 질문과 진단이 끝난 뒤 발주 기관에 전달되는 산출물의 형식도 함께 정해집니다.

6.1.1 「무엇이 걸리고 해소에 얼마가 드는가」로 바뀌는 진단 질문

전환 원칙 세 가지

전환 대상에 적용하는 원칙은 다음 세 가지이며, 번호 순서가 대상 한 건에 원칙을 적용하는 순서입니다.

1. 모든 전환 대상은 컨테이너 전환을 먼저 검토합니다.
2. 가상화(VM) 전환은 6.4절의 예외 사유가 확인된 경우에만 허용합니다. 편의, 일정, 익숙함은 사유에 포함되지 않습니다.
3. 예외로 VM에 남기는 대상에는 확인된 사유, 사유가 해소되는 조건, 재검토 시점을 기록합니다.

첫째 원칙에 따라 컨테이너 전환을 먼저 검토하기로 정하면 사전 진단은 「컨테이너로 옮기는 데 무엇이 걸리고, 그것을 해소하는 데 얼마가 드는가」를 묻게 됩니다. 반면 VM과 컨테이너의 장단점을 견주는 질문에서 출발하면 대상마다 선호와 일정이 판단에 섞이고, 같은 조건의 시스템이 담당자에 따라 다른 방식으로 분류됩니다. 첫째 원칙은 특정 기술에 대한 선호를 밝힌 것이 아니며, 모든 대상에 같은 질문을 적용하기 위한 판정 절차의 출발 조건입니다.

선택지 비교표를 대신하는 장애 요인 목록

질문이 바뀌면 산출물도 달라지는데, VM과 컨테이너를 대등하게 놓고 비교하는 진단은 선택지 비교표를 남기고 결정은 다시 발주 기관의 몫으로 돌아옵니다. 컨테이너 전환을 전제로 한 진단은 대상별 장애 요인 목록과 요인별 해소 공수를 남깁니다. 예를 들어 세션을 로컬 메모리에 저장하는 구조, 상용 WAS에 종속된 코드, 지원이 끝난 Java 버전이 목록의 항목이 되고, 항목마다 Story Point와 일수가 붙습니다(3장과 4장). 의사결정권자는 이 목록을 보고 예산과 기간을 정하며, 해소할 수 없는 요인이 확인된 대상만 둘째 원칙의 예외 심사로 넘어갑니다.

셋째 원칙은 예외가 영구화되는 것을 막기 위한 것으로, 예외로 분류된 VM은 전환을 마친 대상으로 집계하지 않고 컨테이너 전환 전의 대기 상태로 관리합니다. 예외 대상마다 사유가 언제 어떤 조건에서 해소되는지를 적고, 그 시점에 다시 검토합니다. 세 원칙을 함께 적용하면 예외의 수량은 사유 확인의 결과로만 정해지며, 사전에 목표 비율을 두지 않습니다.

6.1.2 IaaS 이전을 시작 단계로 두는 NIA 성숙도 모델의 근거

두 성숙도 모델의 Level 1

NIA의 클라우드 네이티브 성숙도 모델은 2021년 발주자 안내서와 2025년 5월 클라우드 네이티브 구축·운영 가이드에 각각 실려 있습니다. 두 모델은 단계 수가 달라서 2021년 모델은 Level 0부터 Level 3까지 4단계이고, 2025년 모델은 Level 5까지 6단계입니다. 그러나 IaaS 이전의 위치는 두 모델에서 같습니다. 2021년 안내서는 Level 1을 클라우드 준비단계(IaaS)로, 2025년 가이드는 클라우드 시작 단계(Entry)로 두고 VM 기반 전환을 그 내용으로 적었습니다[S43][S45].

단계	2021년 발주자 안내서	2025년 구축·운영 가이드
Level 0	기존 애플리케이션(온프레미스, 모놀리식)	기존 애플리케이션(Initial)
Level 1	클라우드 준비단계(IaaS, 모놀리식)	클라우드 시작 단계(Entry): IaaS 전환, VM 기반
Level 2	클라우드 친화단계(PaaS와 컨테이너, 모놀리식)	클라우드 활용 단계(Operate): PaaS 도입
Level 3	클라우드 네이티브단계(MSA)	클라우드 네이티브 도입 단계 (Scale): MSA, CI/CD, 컨테이너, 데브옵스, 비핵심 시스템부터
Level 4~5	없음	최적화 단계(Optimize), 혁신 단계 (Innovate)

표의 출처는 NIA 발주자 안내서와 구축·운영 가이드입니다[S43][S45]. 표에서 PaaS(Platform as a Service)는 실행 환경까지 제공받아 쓰는 서비스 형태를 가리킵니다.

Level 3 목표와 VM 이전의 거리

VM에서 VM으로 옮기는 이전은 두 모델 어디에서도 Level 1을 넘지 못합니다. 정부의 클라우드 네이티브 전환 상세설계 사업과 본사업 제안요청서는 목표를 Level 3으로 적습니다[S36][S41]. 따라서 VM 이전으로 사업을 마친 시스템은 발주 문서가 요구하는 목표까지 두 단계가 남으며, 그 차이를 메우는 사업을 다시 발주해야 합니다. 제안요청서의 Level 3이 4단계 모델과 6단계 모델 가운데 어느 쪽을 가리키는지에 따라 요구 범위가 달라지므로, 발주 문서에는 기존 모델을 함께 밝혀야 합니다. 두 모델의 차이가 발주 문서에 미치는 영향은 7장에서 다룹니다.

6.1.3 컨테이너에서 데브옵스·CI/CD, MSA로 이어지는 NIA 도입 순서

컨테이너가 첫 순서인 NIA 도입 순서

NIA 발주자 안내서는 클라우드 네이티브 구성요소의 도입 순서를 컨테이너, 데브옵스 및 CI/CD(지속적 통합·배포), MSA 순으로 제시합니다. 같은 안내서의 적합성 검토 체크리스트는 10문항 가운데 「예」가 4개인 기관에 기존 아키텍처를 유지하되 필요하면 구성요소 일부를 도입할 수 있다고 판정합니다[S43]. 적합성 점수가 낮아 MSA를 이루는 기관에도 컨테이너는 부분 도입 대상으로 남습니다. 컨테이너 우선 원칙은 이 순서의 첫 단계를 모든 전환 대상의 검토 출발점으로 삼은 것입니다.

컨테이너 전환과 서비스 분리의 구분

컨테이너 전환과 MSA 전환은 별개의 결정이며, 모놀리식 애플리케이션을 나누지 않고 그대로 컨테이너 이미지로 만들어도 얻는 것이 있습니다. NIA 안내서의 비교표에 따르면 컨테이너는 크기가 MB 단위이고 즉시 기동되며, VM은 GB 단위이고 기동에 수 분이 걸립니다[S43]. 이미지 하나로 개발 환경과 운영 환경을 같게 만들 수 있으므로 환경 차이에서 생기는 장애도 줄어듭니다. 서비스 분리의 타당성이 확인되지 않은 시스템도 컨테이너 전환은 먼저 진행할 수 있고, 분리 여부는 그다음 사업에서 정하면 됩니다.

2025년 모델에 없는 모놀리식 컨테이너 단계

두 성숙도 모델은 이 도입 순서를 다르게 반영하는데, 2021년 모델의 Level 2는 「PaaS와 컨테이너, 모놀리식」으로, 모놀리식 애플리케이션의 컨테이너 전환을 독립된 단계로 둡니다[S43]. 2025년 모델에는 이에 해당하는 독립 단계가 없고, 컨테이너가 Level 3에서 MSA, 데브옵스와 함께 나옵니다[S45]. 따라서 2025년 모델만 기준으로 삼으면 MSA를 적용하지 않는 컨테이너 전환이 어느 단계에 해당하는지 문서로 판정하기 어렵습니다. 사전 진단에서는 대상별로 컨테이너 전환과 서비스 분리를 나누어 판정하고, 각각의 공수를 따로 제시합니다.

6.2 컨테이너 전환과 VM 이전이 남기는 기술 부채 비교

VM 이전은 컨테이너 전환보다 기간이 짧고 초기 비용이 낮으며, 이 사실 때문에 「VM 이전이 더 빠르고 저렴하다」는 의견이 전환 사업마다 제기됩니다. 그러나 두 방식은 옮기는 단위가 다르고, 그 차이가 사업이 끝난 뒤 남는 기술 부채의 크기를 정합니다. VM 이전에서 줄어든 비용은 없어진 비용이 아니며, 같은 시스템의 다음 사업으로 이월된 현대화 비용입니다. 아래 비교에서 VM 이전은 예외 경로의 특성으로만 다룹니다.

6.2.1 게스트 OS를 포함한 머신 이미지와 애플리케이션 단위 이미지의 차이

옮기는 단위의 차이

VM 이전은 게스트 OS(가상 머신 안에 설치된 운영체제)를 포함한 머신 이미지를 통째로 옮깁니다. 컨테이너 전환은 애플리케이션과 실행에 필요한 바이너리, 라이브러리만 이미지로 만듭니다. 컨테이너는 호스트의 리눅스 커널을 다른 컨테이너와 공유하므로 이미지 안에 게스트 OS가 없습니다[S01]. VM은 하이퍼바이저(물리 서버 위에서 여러 VM을 실행하는 가상화 소프트웨어) 위에서 VM마다 운영체제를 따로 실행하며, NIA 안내서는 이 구조에서 생기는 성능 저하를 VM의 특성으로 적었습니다[S43].

두 방식의 차이를 항목별로 견주면 다음 표와 같으며, 표의 수정 범위와 남는 부채는 사전 진단이 대상별로 확인하는 항목입니다.

항목	컨테이너 전환(기본)	VM 이전(예외 경로)
옮기는 단위	애플리케이션과 실행에 필요한 바이너리·라이브러리	게스트 OS를 포함한 머신 이미지

항목	컨테이너 전환(기본)	VM 이전(예외 경로)
게스트 OS	없음. 호스트 리눅스 커널 공유	VM마다 유지
애플리케이션 수정	상태 외부화, 설정의 환경 변수화, 표준 출력 로그, 정상 종료 처리	없음
얻는 것	변경되지 않는 이미지, 빠른 기동, 높은 집적도, OS 패치와 애플리케이션의 분리	인프라 교체
남는 부채	모놀리식 구조의 결합도(서비스 분리하는 다음 단계)	게스트 OS 중복, 구성 편차, 구버전 OS·미들웨어
기간과 비용	수정 범위에 비례. 사전 진단이 산정	짧고 낮음. 현대화 비용은 다음 사업으로 이월

게스트 OS 수와 운영 비용

운영 비용의 차이는 게스트 OS의 수에서 생기는데, VM 100대를 이전하면 패치와 구성 관리의 대상인 운영체제도 100개가 그대로 남습니다. 컨테이너로 전환하면 운영체제 패치 대상은 컨테이너를 실행하는 호스트 노드로 줄고, 애플리케이션은 이미지를 다시 만들어 교체하는 방식으로 갱신합니다. 이렇게 되면 운영체제 보안 패치와 애플리케이션 배포가 서로 다른 주기로 분리됩니다. 다만 성능이나 집적도를 배수로 표현한 수치는 측정 조건에 따라 크게 달라지므로, 여기서는 공개 문서가 밝힌 크기 단위와 기동 시간의 차이만 근거로 삼았습니다 [S43].

가상화 제품을 바꾸는 VM 간 이전

가상화 제품을 다른 제품으로 바꾸는 VM 간 이전도 위 표의 VM 이전과 같은 부채를 남깁니다. 하이퍼바이저가 바뀌어도 VM마다 게스트 OS를 두는 구조는 그대로이므로, 게스트 OS 중복과 구성 편차, 구버전 OS·미들웨어가 새 가상화 환경으로 함께 옮겨집니다. 따라서 가상화 제품의 교체를 계기로 전환을 검토하는 기관도 대상별로 컨테이너 전환 가능 여부를 먼저 판정하고, VM으로 남기는 대상은 사유가 확인된 예외로 기록합니다.

6.2.2 VM 이전 뒤에도 남는 구버전 OS·미들웨어 지원 종료 문제

수정 없는 이전이 남기는 과제

VM 이전에서 애플리케이션을 수정하지 않는다는 것은 이전 방식의 정의입니다. AWS Prescriptive Guidance는 Rehost를 애플리케이션을 전혀 변경하지 않고 옮기는 전략으로 정의합니다[S33]. 수정이 없으므로 이전 전의 운영체제, Java 버전, WAS, 프레임워크가 이전 후에도 같은 버전으로 남습니다. 5장에서 본 지원 종료 일정은 VM 이전을 마친 뒤에도 그대로 유효합니다. 예를 들어 CentOS Linux 8은 2021년 12월 31일에 지원이 끝났습니다[S10]. 이 운영체제를 게스트 OS로 실은 VM은 새 인프라에서도 지원이 끝난 상태로 운영됩니다.

VM 이전을 마친 뒤에도 발주 기관에 남는 과제는 다음 네 가지로 정리됩니다.

- VM마다 유지되는 게스트 OS의 패치와 라이선스
- VM별로 달라진 설정을 수작업으로 맞추는 구성 편차 관리

- 세션 저장 방식, 하드코딩 경로 같은 코드와 구조의 문제
- 지원이 끝난 운영체제, Java, WAS, 프레임워크의 교체

다음 사업으로 이월되는 현대화 비용

NIA의 2025년 가이드는 지원이 종료된 버전을 클라우드 서비스 제공자가 제공하는 최신 안정화 버전으로 올리고, 라이선스를 재활용할 수 없으면 공개 소프트웨어로 플랫폼을 변경해 최신 버전을 적용하도록 방침을 정했습니다[S45]. 정부 제안요청서도 운영체제와 시스템 소프트웨어를 보안이 검증된 최신 안정화 버전으로 올려 전환하도록 요구합니다[S36]. VM 이전으로 사업을 마친 시스템은 이 방침을 충족하기 위한 현대화 사업을 한 번 더 거쳐야 합니다. VM 이전의 낮은 비용은 현대화 비용이 빠진 금액이고, 그 비용은 같은 시스템을 대상으로 한 다음 발주에 다시 나타납니다. 의사결정권자가 두 방식의 비용을 견줄 때는 이월되는 비용을 VM 이전 쪽에 더해야 같은 범위의 비교가 됩니다.

6.2.3 컨테이너 전환을 Replatform으로 고정하는 용어 기준

체계마다 다른 전환유형 용어

전환유형을 가리키는 Rehost, Replatform, Refactor는 체계마다 뜻이 다릅니다. Gartner의 초기 분류를 전한 2011년 보도에서 Refactor는 PaaS로의 재배치를 뜻했고, AWS의 7R에서 Refactor는 클라우드 네이티브 기능을 활용하도록 아키텍처를 고치는 것을 뜻합니다[S32][S33]. 같은 낱말을 발주 기관과 사업자가 다른 뜻으로 쓰면 제안요청서의 과업 범위와 제안서의 수행 범위가 어긋납니다. 전환 원칙을 적용하기 전에 컨테이너 전환이 어느 유형에 해당하는지를 먼저 고정해야 합니다.

체계	컨테이너화가 속하는 유형	정의 요지
AWS Prescriptive Guidance(7R)	Replatform	코드 변경 없이 VM을 컨테이너로 옮겨 성능을 개선하는 경우를 예로 제시
Microsoft Cloud Adoption Framework	Replatform	최소한의 코드 변경으로 워크로드를 현대적인 호스팅 환경으로 이전. 지표에 애플리케이션 컨테이너화 포함
정부 제안요청서(2026년 상세설계 2차)	Replatform(플랫폼전환)에 가까움	OS·WEB·WAS·DBMS 등 플랫폼 변경으로 애플리케이션 소스 수정과 데이터 변환이 수반되는 유형

표의 출처는 AWS Prescriptive Guidance, Microsoft Cloud Adoption Framework, 2026년 클라우드 네이티브 전환 상세설계 사업(2차) 제안요청서입니다[S33][S34][S36].

컨테이너 전환의 유형은 Replatform

컨테이너 전환은 Replatform(플랫폼전환)으로 고정해 씁니다. 세 체계가 모두 컨테이너화를 Replatform으로 분류하거나 그에 가깝게 정의하므로, 공공 발주 문서와 민간 클라우드 사업자의 문서를 함께 읽는 독자에게 혼동이 없습니다. 정부 제안요청서의 Rehost(단순전환)는 시스템 소프트웨어 버전 상향 정도의 변경만으로 전환할 수 있는 유형을 가리킵니다[S36]. 이 기준으로 전환 원칙을 옮기면 'Replatform이 기본이고, VM으로의 Rehost는 예외'가 됩니다. 발주 문서에 전환유형별 대상 수를 적을 때도 같은 기준을 쓰면 제안서의 범위와 견주기 쉽습니다.

6.3 리눅스 실행 가능성으로 정하는 컨테이너 전환 판정 기준

컨테이너 전환 판정 기준은 대상 애플리케이션을 컨테이너로 옮길 수 있는지를 가리는 규칙입니다. 전환 사업에서는 대상마다 「이 시스템은 특수해서 컨테이너로 옮길 수 없다」는 의견이 나오고, 기준이 없으면 의견의 수만큼 예외가 생깁니다. 판정 기준은 「리눅스에서 실행할 수 있게 만들 수 있으면 컨테이너로 간다」는 한 문장입니다. 이 기준은 Docker의 공식 문서가 서술한 컨테이너의 정의에서 끌어낸 결론이며, 단서 세 가지가 붙습니다. 기준이 한 문장으로 정해지면 대상별 논쟁은 단서와 예외 사유를 확인하는 절차로 바뀝니다.

6.3.1 호스트 커널을 공유하는 프로세스라는 컨테이너의 정의

공식 문서가 서술한 컨테이너 정의

판정 기준의 근거가 되는 공식 문서의 서술은 Docker의 컨테이너 정의이며, Docker는 컨테이너가 운영체제 커널을 다른 컨테이너와 공유하고 각각 사용자 공간에서 격리된 프로세스로 실행된다고 설명합니다. 원문 표현은 "share the OS kernel with other containers, each running as isolated processes in user space"입니다[S01]. 커널을 공유하므로 컨테이너는 호스트의 리눅스 커널에 직접 의존합니다. 이 서술에 따르면 컨테이너는 가상의 하드웨어나 별도의 운영체제를 갖지 않으며, 호스트 리눅스 커널 위에서 실행되는 격리된 프로세스입니다.

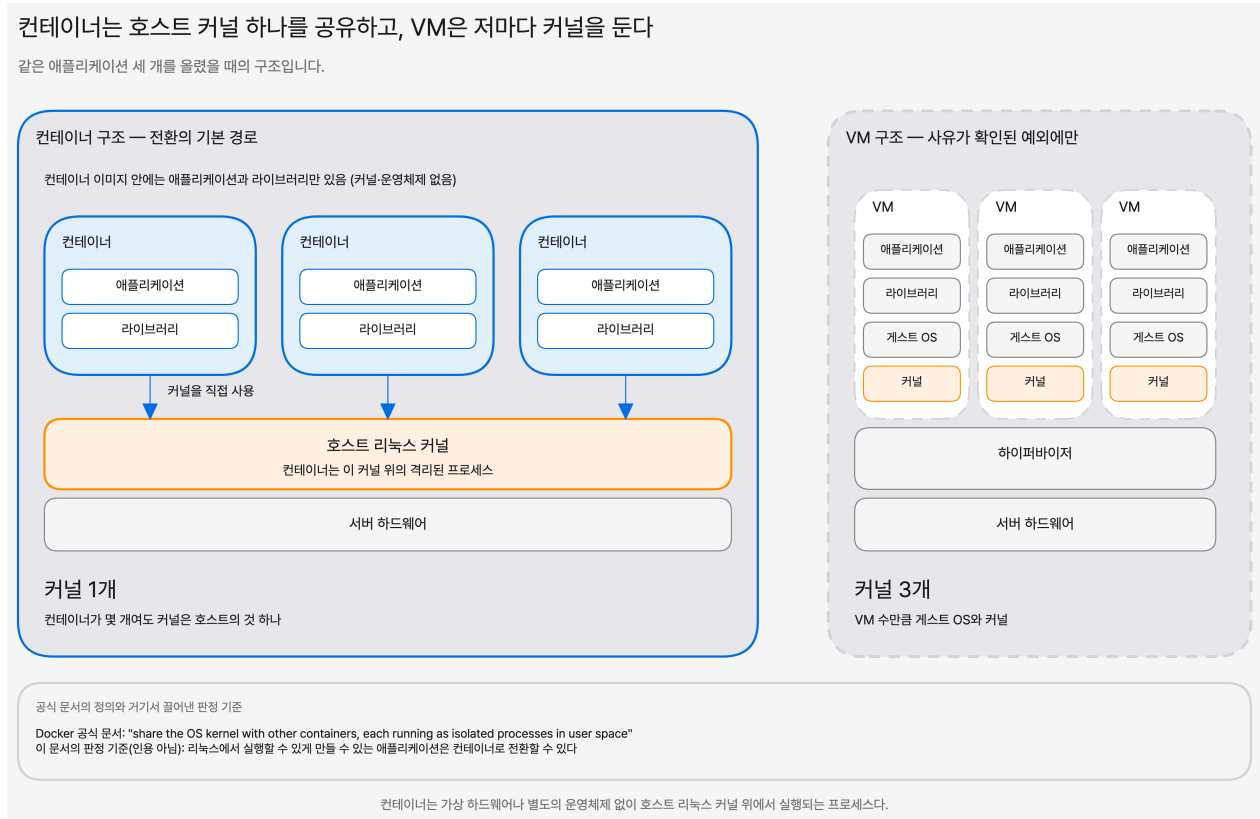


그림. 호스트 리눅스 커널 하나를 여러 컨테이너가 공유하는 구조와 VM마다 게스트 OS를 두는 구조

커널 공유 정의에서 끌어낸 판정 기준

여기서부터는 공식 문서에 적힌 서술이 아니며, 그 서술에서 끌어낸 결론입니다. 컨테이너가 리눅스 커널 위의 프로세스라면, 리눅스에서 프로세스로 실행할 수 있는 애플리케이션은 컨테이너로도 실행할 수 있습니다. 따라서 「리눅스에서 실행할 수 있게 만들 수 있는 애플리케이션은 컨테이너로 전환할 수 있다」가 판정 기준이 됩니다.

Docker의 문서 어디에도 이 명제가 한 문장으로 적혀 있지는 않습니다. 위 서술과 6.3.2항의 단서 세 가지를 함께 놓았을 때 성립하는 추론이므로, 발주 문서나 내부 기준에 옮겨 적을 때도 근거 서술과 결론을 구분해 적어야 합니다.

이 기준에서 「실행할 수 있게 만들 수 있으면」이라는 표현에는 수정과 포팅이 포함됩니다. 현재 Unix나 Windows에서 실행 중인 애플리케이션도 리눅스에서 실행되도록 고칠 수 있으면 컨테이너 전환 대상입니다. 판정의 쟁점은 컨테이너 기술의 적합성에서 리눅스 전환 가능성과 그 공수로 옮겨 가고, 이 공수는 사전 진단의 정적 분석과 체크리스트로 산정할 수 있습니다.

6.3.2 커널 모듈·이미지와 호스트 버전 차이·CPU 아키텍처 세 가지 단서

판정 기준에 붙는 단서 세 가지

리눅스에서 실행된다는 사실만으로 컨테이너 전환이 보장되지 않는 경우가 세 가지 있습니다. 각 단서의 근거와 확인 방법은 다음과 같습니다.

1. 커널에 직접 종속된 것은 제외됩니다. 컨테이너는 커널 모듈을 스스로 적재할 수 없고[S02], 커널 버전별 모듈이나 자료구조와 직접 상호작용하는 애플리케이션은 호스트 커널이 바뀌면 동작을 보장할 수 없습니다. 대상 서버에 설치된 커널 모듈과 장치 목록으로 확인합니다.
2. 이미지와 호스트의 메이저 버전 차이에는 지원 한계가 있습니다. 한계의 폭은 배포판 공급자마다 다르므로 사용 중인 배포판의 호환 기준을 따릅니다. 현재 운영체제 버전과 목표 플랫폼의 호스트 버전을 대조해 확인합니다.
3. CPU 아키텍처가 같아야 합니다. POWER, SPARC, Itanium용으로 빌드한 Unix 바이너리는 x86 리눅스에서 실행되지 않습니다. 바이너리는 특정 CPU 명령어 집합과 커널, C 라이브러리에 맞춰 빌드되기 때문입니다. 대상 장비의 CPU 종류와 소스 보유 여부로 확인합니다.

셋째 단서의 결론도 공식 문서에 명시된 문장이 아니며 추론에 해당합니다. AIX, HP-UX, Solaris 바이너리가 리눅스 컨테이너에서 실행되지 않는다고 직접 적은 공개 문서는 확인되지 않았고, 근거는 바이너리가 빌드 대상 CPU 아키텍처와 커널·C 라이브러리에 종속된다는 일반 원리입니다.

구버전 사용자 공간 이전의 짧은 지원 기간

둘째 단서는 현대화를 병행해야 하는 이유와 이어지는데, 구버전 운영체제에서 실행되던 애플리케이션을 수정하지 않으려고 그 운영체제의 사용자 공간을 그대로 이미지로 만들어 옮기는 방식이 있습니다. 배포판 공급자는 이미지와 호스트의 메이저 버전 차이에 지원 한계를 두므로, 구버전 운영체제 사용자 공간을 그대로 이미지로 옮기는 방식은 오래 유지하기 어렵습니다. 구버전 운영체제의 사용자 공간을 통째로 옮기는 컨테이너 전환은 VM 이전과 같은 부채를 이미지 안에 남기며, 유효 기간도 호스트 운영체제의 버전 상향 주기에 묶입니다. 따라서 컨테이너 전환에서는 이미지의 기반 운영체제와 Java, WAS 버전을 함께 올리는 것이 원칙이고, 그 공수는 5장에서 다룬 현대화 항목으로 산정합니다.

6.3.3 Windows 컨테이너로 전환 가능한 서버형 Windows 애플리케이션

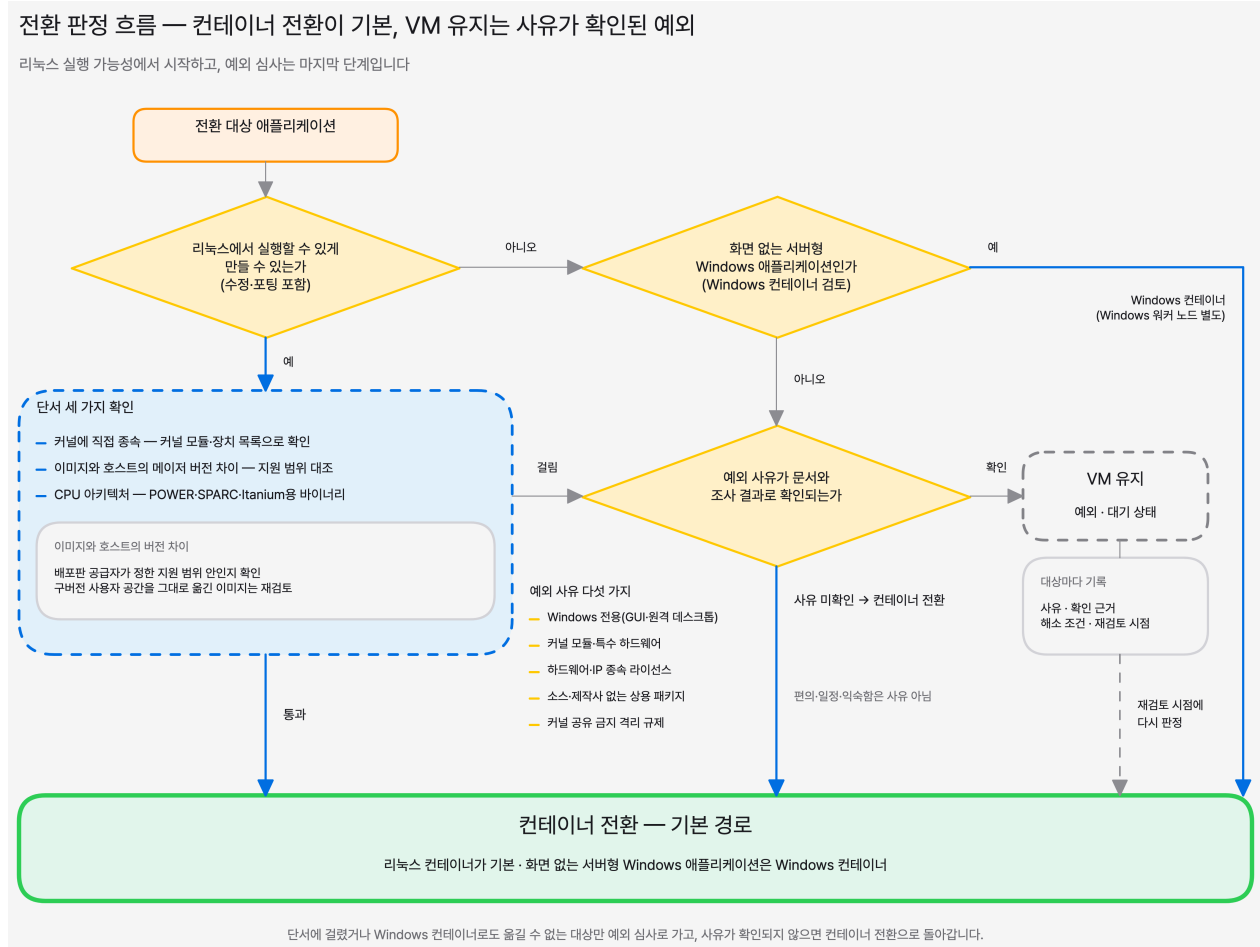
Windows 워크로드의 검토 순서

Windows 전용 소프트웨어는 Windows에서 실행된다는 사실만으로 가상화 예외가 되지 않습니다. 검토 순서는 세 단계이며, 먼저 리눅스로 전환할 수 있는지를 확인하고, 전환할 수 있으면 리눅스 컨테이너로 옮깁니다. 리눅스 전환이 어려우면 Windows 컨테이너를 검토합니다. 화면이 없는 서버형 Windows 애플리케이션은

Windows 컨테이너로 전환할 수 있습니다[S04]. 두 단계를 모두 통과하지 못한 대상만 가상화 예외 심사로 넘어갑니다.

Windows 컨테이너의 운영 조건과 예외 경계

Kubernetes는 Windows 워커 노드를 지원합니다[S03]. 워커 노드는 컨테이너가 실제로 실행되는 서버를 가리키며, Windows 컨테이너는 리눅스 노드에서 실행되지 않으므로 클러스터에 Windows 워커 노드를 별도로 두어야 합니다. 따라서 Windows 컨테이너를 선택하면 리눅스 노드와 Windows 노드를 함께 운영하는 부담이 생기고, 사전 진단에서는 이 부담을 대상 수와 함께 제시합니다. 데스크톱·GUI 애플리케이션, 원격 데스크톱에 의존하는 애플리케이션, Windows Server 2016 미만에서만 실행되는 애플리케이션은 Windows 컨테이너로 옮길 수 없으므로 예외 사유에 해당합니다[S04].



6.4 가상화 전환을 허용하는 예외 사유와 기록 항목

가상화 전환을 허용하는 예외 사유는 컨테이너로 옮길 수 없음이 확인된 경우와, 리눅스에서 실행되더라도 커널 공유가 허용되지 않는 경우로 한정됩니다. NIA의 2025년 가이드는 전환유형을 고르는 정량 기준을 제시하지 않고, 사업자가 기준과 절차를 수립하도록 넘깁니다[S45]. 기준이 없는 사업에서는 예외의 범위가 사업자마다 달라지고, 발주 기관은 VM으로 남은 대상의 타당성을 검증하기 어렵습니다. 아래 예외 사유표는 사전 진단의 판정 기준이며, 발주 문서에 첨부해 제안서의 VM 수량을 검증하는 근거로 쓸 수 있습니다.

6.4.1 가상화 전환을 허용하는 예외 사유와 사유별 확인 절차

가상화 예외 사유 다섯 가지

가상화 예외 사유는 다섯 가지이며, 사유마다 확인 방법과 먼저 시도할 해소 경로가 있습니다. 해소 경로를 시도한 뒤에도 사유가 남은 대상만 VM으로 유지합니다.

예외 사유	확인 방법	먼저 시도할 해소 경로	해소되지 않을 때
리눅스나 Windows 컨테이너로 전환할 수 없는 Windows 전용 소프트웨어	화면(GUI) 유무, .NET Framework 버전, OS 버전	리눅스 전환 후 컨테이너화, 서버형이면 Windows 컨테이너	데스크톱·GUI, 원격 데스크톱 의존, Windows Server 2016 미만은 VM 유지
커널 모듈·특수 하드웨어·실시간 OS 의존	설치된 커널 모듈과 장치 목록	사용자 공간 대체 수단, 전용 노드	VM 유지
하드웨어나 IP에 종속된 라이선스	라이선스 계약과 인증 방식	라이선스 교환, 전용 노드 고정	VM 유지
소스가 없거나 제작사가 없는 상용 패키지	소스 보유 여부, 제작사 존속 여부	제작사의 컨테이너 지원판, 대체 소프트웨어	VM 유지
커널 공유를 금지하는 격리 규제	적용 규정의 조문	전용 노드, 클러스터 분리	VM 유지

이 사유들은 공개 가이드의 서술과 겹칩니다. Google Cloud의 Migrate to Containers 계획 가이드는 특수 커널 드라이버를 쓰는 VM, 특정 하드웨어 의존, 하드웨어 식별자 등록에 종속된 라이선스, 데스크톱·원격 데스크톱 애플리케이션을 컨테이너에 부적합한 대상으로 들고, Tomcat 같은 웹·애플리케이션 서버와 미들웨어를 적합한 대상으로 듭니다[S05]. NIA의 2025년 가이드도 소프트웨어 제작업체가 없는 경우, 현재 버전의 소스 코드를 보유하지 않은 경우, 상용 어플라이언스를 쓰는 경우를 전환 불가(Retain) 예시로 적었습니다[S45].

사유가 확인되지 않은 대상의 처리

사유가 확인되지 않은 대상은 예외가 될 수 없습니다. 편의, 촉박한 일정, 운영 조직의 익숙함은 표의 어느 사유에도 해당하지 않으므로 예외 근거로 인정하지 않습니다. 사유의 판정 근거는 담당자의 의견을 제외한 문서와 조사 결과이며, 예를 들어 「라이선스 때문에 옮길 수 없다」는 의견은 계약서의 인증 방식 조항과 제작사 회신으로 확인하고, 확인 근거가 없으면 컨테이너 전환 대상으로 분류합니다.

가상화 예외에 포함되지 않는 Unix 바이너리

리눅스판이 없는 Unix 바이너리는 가상화 예외와 구분해 따로 다루는데, AIX, HP-UX, Solaris용 바이너리는 6.3.2항의 셋째 단서와 같은 이유로 x86 VM에서도 실행할 수 없기 때문입니다. CPU 아키텍처가 다른 운영체제는 x86 하이퍼바이저의 게스트로 올릴 수 없다는 것이 이 백서의 추론입니다. 따라서 이 대상의 선택지는 리눅스 포팅 후 컨테이너 전환, 대체 소프트웨어 도입, 기존 장비 유지 세 가지이며, VM 유지는 선택지에 들어가지 않습니다. 포팅 공수는 5장의 OS 전환 항목으로 산정합니다.

6.4.2 예외가 아닌 수정 대상: 세션·하드코딩 경로·구버전 Java와 Unix 위의 Java 애플리케이션

수정 대상과 전환 불가 대상의 구분

컨테이너 전환에서 수정이 필요하다는 것과 전환할 수 없다는 것은 서로 다른 판정이며, 전환 사업에서 예외로 자주 제기되는 항목 가운데 다수는 수정 대상에 해당합니다. 다음 항목은 가상화 예외가 아니며, 컨테이너로 전환하면서 수정합니다.

- 로컬 디스크에 상태를 저장하는 구조, 대용량 메모리 세션, IP를 지정하는 세션 클러스터링
- 호스트명이나 IP에 의존하는 로직, 하드코딩된 파일 경로
- 수십 분이 걸리는 기동 시간, root 권한 실행, 배치 작업
- 구버전 Java, WAS, 프레임워크
- Unix에서 실행 중인 Java 애플리케이션

앞의 세 묶음은 Twelve-Factor App 원칙이 설정의 외부화, 상태 없는 프로세스, 빠른 기동과 정상 종료로 다루는 항목과 같습니다. 구버전 Java와 WAS는 5장의 현대화 대상이고, 상용 WAS에 종속된 코드는 애플리케이션 현대화 분석 도구가 필수 수정 항목으로 분류해 Story Point를 매깁니다. 이 항목들은 분석 도구와 체크리스트로 견수와 공수가 산정되는 대상입니다.

Unix 위의 Java 애플리케이션

Unix에서 실행 중인 Java 애플리케이션은 리눅스 전환을 검증할 대상이며 예외로 분류하지 않습니다. JVM 명세는 JVM이 하드웨어와 운영체제로부터의 독립성을 담당한다고 정의하므로[S20], 순수 Java 애플리케이션은 이식 난이도가 낮은 대상입니다. 다만 JNI로 네이티브 라이브러리를 호출하는 부분은 플랫폼이 바뀌면 다시 빌드해야 하므로 별도 확인 항목으로 둡니다[S21]. 수정 대상을 예외와 구분해 두면 예외 목록이 수정 부담을 이유로 늘어나는 것을 막을 수 있고, 의사결정권자는 예외 목록의 각 항목에 6.4.1항의 사유 가운데 어느 것이 확인됐는지만 검토하면 됩니다.

6.4.3 예외 대상마다 기록하는 사유·해소 조건·재검토 시점

예외 기록 양식의 다섯 열

예외로 VM에 남기는 대상에는 다음 양식으로 기록을 남기며, 표의 세 행은 작성 방법을 보이기 위한 예시입니다.

대상	사유	확인 근거	해소 조건	재검토 시점
상용 패키지 A	제작사의 컨테이너 지원판 없음	제작사 회신 문서	지원판 출시 또는 대체 소프트웨어 선정	다음 유지보수 계약 갱신 시
장비 연동 모듈 B	전용 커널 모듈 의존	설치 모듈 목록	사용자 공간 드라이버 제공	장비 교체 사업 착수 시
업무 시스템 C의 관리 화면	Windows 데스크톱 GUI	화면 구성과 OS 버전 조사	웹 화면으로 재개발	차기 고도화 사업 기획 시

NIA의 2025년 가이드는 담당자가 사업자의 전환유형 검토에 「누구나 수궁할 수 있는 체계적, 정량적인 검토 기준」이 제시됐는지 확인하도록 요구합니다[S45]. 이 양식의 사유와 확인 근거 열이 그 검토 기준에 해당하며, 제3자가 같은 근거로 같은 판정에 도달할 수 있어야 합니다.

재검토 시점과 로드맵의 연결

기록이 없으면 예외는 영구화되는데, 사유를 적지 않은 VM은 몇 년 뒤 담당자가 바뀌면 왜 VM으로 남았는지 확인할 수 없고, 그대로 다음 인프라 교체 사업의 이전 대상이 됩니다. 예외 대상은 전환 완료로 집계하지 않고 컨테이너 전환 전의 대기 상태로 관리합니다. 재검토 시점은 전환 로드맵의 일정 항목이 되며, 예를 들어 유지보수 계약 갱신, 장비 교체, 고도화 사업 기획처럼 사유가 해소될 가능성이 있는 시점을 적어 두면, 그 시점에 해소 조건을 다시 확인하고 컨테이너 전환 여부를 판정할 수 있습니다. 진단 이후의 로드맵 구성은 8장에서 다룹니다.

6.4.4 비용 초과를 가상화 예외가 아닌 전환 방식 재결정 사유로 다루는 원칙

NIA 가이드의 비용 기준과 적용 방식

NIA의 2025년 가이드는 전환 비용이 초기 구축 비용의 50%를 넘는 경우를 전환 불가 판정의 예시로 듭니다 [S45]. 이 기준은 전환에 드는 비용이 지나치게 큰 대상을 가려내는 데 쓸 수 있습니다. 다만 컨테이너 우선 원칙에서는 비용 초과만으로 가상화 예외를 인정하지 않습니다. 6.4.1항의 다섯 사유는 모두 기술이나 계약, 규제로 확인되는 사실이고, 비용은 예산 사정에 따라 달라지는 판단이기 때문입니다. 비용을 예외 사유에 넣으면 같은 시스템이 예산이 넉넉한 해에는 컨테이너 전환 대상이 되고 부족한 해에는 예외가 됩니다.

비용 초과 대상의 세 가지 선택지

비용이 기준을 넘는 대상은 전환 방식을 다시 정하는 절차로 보내며, 선택지는 재개발, 단계 분할, 폐기 세 가지입니다.

선택지	적용 조건	결정에 필요한 진단 결과
재개발	수정 공수가 새로 개발하는 공수에 가깝거나 넘는 경우	필수 수정 항목의 Story Point 합계
단계 분할	공수는 크지만 업무 단위로 나눠 옮길 수 있는 경우	업무별 장애 요인과 의존 관계
폐기	이용이 적고 다른 시스템으로 기능을 대체할 수 있는 경우	이용 현황과 대체 가능 기능

비용 초과는 VM으로 옮길 근거가 되지 않으며, 옮기는 방식을 바꿀 근거가 됩니다. 기술 사유와 예산 판단을 분리하면 예외 목록은 예산 편성과 무관하게 유지되고, 의사결정권자는 비용 초과 대상만 따로 모아 예산 심의 안건으로 다룰 수 있습니다.

6.5 예외로 남은 VM을 컨테이너 플랫폼에서 함께 운영하는 구성

예외 사유가 확인된 VM은 사유가 해소될 때까지 운영해야 하며, 이 VM을 위해 기존 가상화 기반을 따로 유지하면 컨테이너 플랫폼과 가상화 플랫폼을 이중으로 운영하게 되고, Kubernetes 위에서 VM을 함께 실행하면 플랫폼은 하나로 줄어듭니다. 의사결정권자는 이 가운데 어느 구성으로 예외 대상을 운영할지 정해야 합니다. 다만 KubeVirt처럼 VM을 Kubernetes 위에서 실행하는 기술은 예외 대상을 임시로 받는 수단에 한정되며, 전환 경로로 쓰이면 6.1절의 원칙이 무너집니다.

6.5.1 KubeVirt의 위치: 컨테이너 전환 전까지 예외 VM을 받는 브릿지 기술

KubeVirt에 대한 이 백서의 규정

KubeVirt는 브릿지 기술입니다. 이 백서는 KubeVirt를 예외 사유가 해소돼 컨테이너로 옮기기 전까지 예외 VM을 임시로 받는 수단으로 규정합니다. 따라서 KubeVirt는 전환의 목표 구성이 될 수 없고, 권장 기술도 될 수 없습니다. KubeVirt 위에서 실행되는 VM의 수는 전환 성과로 집계하지 않으며, 줄여 나가야 할 잔여 대상의 수로 관리합니다.

이 규정은 KubeVirt 프로젝트가 스스로 밝힌 적용 대상과 일치합니다. KubeVirt는 Kubernetes에서 VM을 실행하는 오픈소스 기술이며, 공식 소개는 대상을 "existing Virtual Machine-based workloads that cannot be easily containerized", 즉 쉽게 컨테이너화할 수 없는 기존 VM 기반 워크로드로 한정합니다[S06]. 컨테이너로 옮길 수 있는 워크로드는 처음부터 KubeVirt의 적용 대상에 들어 있지 않습니다.

권장 기술이 될 수 없는 근거

VM을 그대로 옮기면 게스트 OS와 구버전 미들웨어의 부채가 그대로 남습니다. KubeVirt 위의 VM도 VM이므로 6.2.2항에서 본 지원 종료 문제, 게스트 OS 패치, 구성 편차가 실행 위치만 바뀐 채 유지됩니다. KubeVirt는 6.4.1항의 예외 사유가 확인된 대상에만 쓰고, 대상마다 6.4.3항의 양식으로 컨테이너 전환 재검토 시점을 함께 기록합니다. 받을 수 있는 게스트 운영체제도 제한됩니다. VM은 호스트와 같은 CPU 아키텍처의 게스트만 실행하므로, x86 플랫폼에서는 POWER·SPARC·Itanium용 Unix 워크로드를 이 수단으로도 받을 수 없습니다. 이 결론은 6.3.2항의 셋째 단서에서 끌어낸 이 백서의 추론입니다.

파드와 VM이 공유하는 운영 인터페이스

예외 VM을 Kubernetes 위에서 실행하면 운영 인터페이스의 상당 부분을 파드(Pod, Kubernetes가 컨테이너를 배치하고 관리하는 최소 단위)와 함께 씁니다. 파드와 VM이 공유하는 주요 운영 인터페이스는 다음 표의 네 가지입니다.

운영 인터페이스	파드와 VM이 함께 쓰는 내용
스케줄러	노드 자원을 기준으로 한 배치
네트워크	클러스터 네트워크와 접근 정책
스토리지	영구 볼륨 요청과 할당
권한 관리	역할 기반 접근 통제

이 공유는 예외 VM 때문에 가상화 플랫폼과 운영 조직을 따로 유지하는 부담을 줄여 줍니다. 그러나 VM 내부의 운영체제와 미들웨어 관리는 공유 범위에 들어가지 않으며, 운영 조직은 파드와 다른 절차로 이를 계속 관리해야 합니다. 운영 인터페이스의 공유는 예외 VM을 받는 기간의 부담을 줄이는 효과로만 평가하고, VM을 유지할 근거로 삼지 않습니다.

6.5.2 기본 경로로 쓸 때 고착되는 파드·VM 이중 운영의 위험

VM 실행 기능을 기본 경로로 쓴 결과

Kubernetes의 VM 실행 기능을 기본 전환 경로로 쓰면 파드와 VM의 이중 운영이 고착됩니다. 수정 없이 옮길 수 있다는 이유로 VM을 먼저 플랫폼에 올리면, 사업은 일정 안에 끝나지만 대상 대부분이 VM으로 남습니다. 플랫폼은 Kubernetes로 바뀌어도 애플리케이션은 이전과 같은 방식으로 배포되고 패치됩니다. 빠른 기동, 이미

지 단위 배포, 높은 집적도 같은 컨테이너 전환의 효과는 파드로 옮긴 일부 대상에서만 나타나고, 운영 조직은 두 가지 운영 절차를 계속 유지해야 합니다.

공공 본사업에 나타난 노드와 VM의 혼합 구성

공공 본사업의 목표 구성에도 파드와 VM이 함께 나타납니다. 2025년에 발주된 한 지방교육청의 클라우드 네이티브 전환 본사업 제안요청서는 목표 구성으로 클러스터 2식과 노드 20식, VM 17식을 제시하고, 재활용하는 상용 소프트웨어는 VM이나 워커 노드에, VOD와 DNS, 접속대기 기능은 VM에 배치합니다[S41]. 이 제안요청서에는 컨테이너화가 어려운 대상을 판정하는 조항이 따로 없고, 해당 대상은 VM 수량으로만 나타납니다. VM 17식이라는 수량이 타당한지는 대상별 사유가 있어야 검증할 수 있습니다.

본사업 전에 정하는 예외 VM 목록

사전 진단은 이 VM 목록을 사유와 함께 본사업 전에 정합니다. 목록에 오르는 대상은 6.4.1항의 사유가 확인된 대상으로 한정되고 사유가 없는 VM은 목록에 넣지 않으며, 다음 표는 특정 사업의 내용과 무관한 작성 예시입니다.

대상	사유	재검토 시점
재활용 상용 소프트웨어	제작사의 컨테이너 지원판 없음	라이선스 갱신 시
영상 전송 장비 제어 서버	특수 하드웨어 의존	장비 교체 시
Windows 데스크톱 관리 도구	GUI 애플리케이션	웹 화면 재개발 사업 기획 시

이 목록에서 VM의 수량은 사유 확인의 결과로만 정해집니다. 따라서 발주 문서에는 VM 비율이나 목표 수량을 먼저 적지 않고, 진단에서 사유가 확인된 대상의 수를 그대로 옮겨 적습니다. 본사업 제안서가 이보다 많은 VM을 제시하면 발주 기관은 추가된 대상의 사유와 확인 근거를 요구할 수 있습니다.

6.5.3 애플리케이션 전환 원칙과 구분하는 데이터 계층 배치

데이터베이스 배치와 가상화 예외의 구분

데이터베이스를 컨테이너 밖의 관리형 서비스나 VM에 두는 구성은 널리 쓰입니다. NIA의 2025년 가이드가 제시한 목표 구성 예시도 데이터베이스를 컨테이너 밖에 두고, Google Cloud의 계획 가이드는 고성능·대용량 인 메모리 데이터베이스를 컨테이너에 부적합한 대상으로 듭니다[S45][S05]. 이 구성은 6.1절 전환 원칙의 예외에 해당하지 않는데, 전환 원칙은 애플리케이션 계층의 전환 방식을 정하는 규칙이고, 데이터베이스를 어디에 둘지는 데이터 계층 배치의 문제로 따로 판정합니다.



그림. 컨테이너로 전환하는 애플리케이션 계층과 따로 판정하는 데이터 계층의 배치 구분

데이터베이스 위치로 생기는 분류 오류

두 계층을 구분하지 않으면 데이터베이스가 VM에 있다는 이유로 그 데이터베이스를 쓰는 애플리케이션까지 가상화 예외로 분류하는 오류가 생깁니다. 애플리케이션은 접속 정보를 환경 변수로 외부화하면 데이터베이스의 위치와 무관하게 컨테이너로 실행할 수 있습니다. 따라서 예외 목록의 대상은 시스템 전체로 묶지 않고 구성요소 단위로 적으며, 데이터베이스가 VM에 남더라도 웹과 WAS는 컨테이너 전환 대상으로 분류합니다.

이 백서의 범위 밖인 데이터베이스 전환 공수

데이터베이스 전환 공수는 애플리케이션 전환 공수와 별도로 산정합니다. NIA의 2025년 가이드에 실린 IaaS 전환용 설계기준 예시는 이기종 데이터베이스 전환에 설치 5 M/D와 이관 12 M/D를, 이에 따른 애플리케이션의 쿼리·프로시저 수정에 17 M/D를 제시합니다[S45]. 이 수치는 서버당 기준의 예시이며 컨테이너 전환용으로 작성된 표는 아닙니다. 데이터베이스 제품 선정과 이관 방식은 사전 진단의 애플리케이션 분석 범위에 포함하지 않으며, 사전 진단에서는 애플리케이션이 데이터베이스에 의존하는 부분, 즉 제품 고유 쿼리와 프로시저 호출의 건수를 애플리케이션 쪽 수정 항목으로 산정합니다.

7장. 정부 상세설계·본사업·NIA 가이드가 비워 둔 자리

6장의 전환 원칙과 예외 사유표는 공공 발주 문서에 실려 있지 않은 판정 기준입니다. 정부는 클라우드 네이티브 전환 본사업에 앞서 상세설계 사업을 따로 발주하고, NIA는 2021년 발주자 안내서와 2025년 구축·운영 가이드로 클라우드 네이티브 전환의 절차를 안내합니다. 2026년 2차 상세설계 사업은 3개 기관 3개 시스템에 약 13억 3천만 원과 200일을 배정했고, 과업 범위는 현황분석부터 후속 사업 제안요청서 도출까지입니다. 이 문서들은 전환유형을 가르는 난이도 등급, 대상 선정 기준, 컨테이너 전환이 어려운 대상의 처리 방침을 직접 정하지 않고 사업자의 과업으로 넘깁니다. 따라서 대상 시스템이 지정되기 전에는 어느 시스템을 올릴지, 컨테이너로 전환할 수 있는지, 공수가 어느 정도인지를 판단할 공공 기준이 없습니다[S36][S45].

사전 진단은 대상이 지정되기 전 단계에서 그 판단을 점수와 일수로 먼저 제시하는 절차이며, 상세설계를 대체하지 않고 그 앞 단계에 놓입니다. 발주 문서와 NIA 자료가 정한 것과 사업자에게 넘긴 것을 구분하면, 본사업 제안요청서의 요구사항에서 사전 진단이 미리 정해 둔 여덟 항목이 나옵니다. 정책결정권자는 이 구분을 근거로 사전 진단을 상세설계 앞 단계에 배치하고, 발주 문서에 넣을 정보가 무엇인지 확인할 수 있습니다.

7.1 상세설계 사업 제안요청서가 사업자에게 넘기는 판단 항목

클라우드 네이티브 전환 상세설계 사업은 본사업의 범위와 예산을 정하기 위해 과학기술정보통신부와 NIA가 발주하는 선행 사업입니다. 제안요청서는 과업의 단계와 산출물, 전환유형의 이름, 비용 산정 방식을 정해 두었습니다. 반면 유형을 가르는 등급과 점수, 대상을 고르는 기준은 사업자가 수립해 제시하도록 했습니다[S36]. 상세설계는 대상 시스템이 지정된 뒤에 시작하므로, 지정 이전의 판단에 쓸 기준은 발주 문서에서 찾을 수 없습니다. 그 결과 발주 기관은 시스템당 수억 원이 드는 단계에 어느 시스템을 올릴지를 별도의 근거 없이 정해야 합니다.

7.1.1 현황분석에서 후속 제안요청서 도출까지 네 단계 과업

상세설계 과업 네 단계

상세설계 과업은 현황분석, 목표모델 수립, 전환 이행계획, 예산산정과 후속 사업 제안요청서 도출의 네 단계입니다. 2026년 2차 제안요청서는 이 단계를 요구사항 네 건으로 적었으며, 그 가운데 현황분석은 조사 서식과 체크리스트 작성, 소스코드 검토를 포함한 전수 조사, 기능점수 현행화를 요구합니다. 목표모델 수립은 전환 방식과 전환유형별 설계 기준, MSA 전환 가능 여부 검토, 컨테이너 용량 산정을 요구합니다. 이행계획과 세부계획은 우선순위와 로드맵, 소요예산 산출, 기능점수 외부 검증, 후속 전환사업 제안요청서 작성을 요구합니다[S36].

2026년 문서에는 2024년 문서에 없던 요구가 추가됐는데, 현황조사에 소스코드 검토를 통한 누락 없는 조사가 들어갔고 클라우드 서비스 제공 위치 검토, 백업·재해복구 예산, 개인정보 검토가 새로 생겼습니다. 기술 평가에서도 수행계획 배점이 15점에서 25점으로 늘었습니다[S36]. 즉 발주 측도 코드 수준의 현황 파악과 수행 방법의 구체성을 이전보다 비중 있게 평가합니다.

연도별 사업 규모

2024년에는 상세설계 사업이 네 차수로 나뉘어 발주됐고, 2026년 2차 사업은 Unix와 구형 상용 소프트웨어가 많은 3개 시스템을 대상으로 삼았습니다. 아래 표의 예산은 각 제안요청서에 기재된 금액을 그대로 옮긴 것입니다.

사업	예산	기간	대상
2024년 1차	1,441,000,000원	계약일~2024-11-29	3개 기관 9개 시스템
2024년 2차	1,167,000,000원	계약일~2024-11-29	3개 기관
2024년 3차	1,675,000,000원	계약일~2024-11-29	3개 기관
2024년 4차	1,349,000,000원	계약일~2024-11-29	3개 기관
2026년 2차	1,328,754,379원(부가세 포함)	계약일로부터 200일	3개 기관 3개 시스템

출처: 각 사업 제안요청서[S36][S37][S38][S39][S40]

2024년 네 차수의 예산을 더하면 약 56억 원이고 대상은 4개 권역 21개 시스템이므로, 단순 평균으로 시스템 당 약 2억 7천만 원이며, 2026년 2차 사업은 같은 방식으로 계산하면 시스템당 약 4억 4천만 원입니다[S35][S36]. 다만 이 값은 예산을 대상 수로 나눈 단순 평균이며, 시스템의 규모와 난이도에 따른 실제 배분은 문서에 나와 있지 않습니다. 상세설계가 시스템당 수억 원 단위의 지출이라는 점은 분명하므로, 어느 시스템을 이 단계에 올릴지에 대한 판단이 먼저 필요합니다.

7.1.2 문서에 없는 난이도 등급표·적합성 점수표·대상 선정 기준

전환유형 세 가지의 정의

상세설계와 본사업 제안요청서는 전환유형을 같은 문구로 정의합니다. Rehost(단순전환)는 시스템 소프트웨어 버전 업그레이드 정도의 변경만으로 클라우드 환경으로 옮길 수 있는 유형입니다. Replatform(플랫폼전환)은 OS, WEB-WAS, DBMS 같은 플랫폼을 바꾸면서 애플리케이션 소스 수정과 데이터 변환이 따르는 유형입니다. Refactor(재구성)는 변경 정도에 따라 일부 개정과 전면 재개발로 나뉘고, 사업기간 안에 끝낼 수 없는 전면 재개발은 제외됩니다[S36][S37][S41].

한 시스템이 세 유형 가운데 어디에 속하는지를 가르는 기준은 「전환시 난이도, 수정의 범위」라는 표현뿐입니다. 난이도를 몇 등급으로 나누는지, 수정 범위를 무엇으로 재는지, 적합성을 몇 점으로 매기어지는 문서에 없고 사업자가 수립할 과업입니다[S36]. NIA의 2025년 가이드도 같은 구조여서, 사업자가 유형 선정의 기준과 절차를 수립하고, 발주 담당자는 누구나 수급할 수 있는 체계적이고 정량적인 검토 기준이 제시됐는지 확인하도록 안내합니다[S45].

기능점수 중심의 비용 산정

비용 산정 방식은 문서가 정해 두었으며, 산정 항목은 애플리케이션 개발비, 클라우드 이용료, 인프라 전환비이고, 개발비의 근거인 기능점수는 제3자 검증을 받습니다. 반면 플랫폼 변경에 따른 소스 수정처럼 기능이 바뀌지 않는 작업의 공수를 어떻게 산정하는지는 기준이 없습니다[S36][S37]. 이 경우 Java 버전 상향이나 상용 WAS 종속 코드 수정에 드는 일수는 사업자의 경험값으로 채워집니다.

구분	정부 문서가 정한 것	사업자에게 넘긴 것
과업 구조	네 단계 과업과 요구사항 네 건	조사 서식과 체크리스트의 구체 항목

구분	정부 문서가 정한 것	사업자에게 넘긴 것
전환유형	Rehost-Replatform-Refactor의 정의	유형을 가르는 난이도 등급표와 적합성 점수표
대상	상세설계 대상 기관과 시스템	지정 이전 단계의 대상 선정 기준
비용	기능점수 기반 개발비와 제3자 검증	기능 변경이 없는 수정 작업의 공수 산정 기준
MSA	전환 가능 여부 검토와 대상 확정 요구	적용할 조건과 적용하지 않을 조건

출처: 2026년 2차 상세설계 사업 제안요청서와 NIA 구축·운영 가이드[S36][S45]

이 표의 오른쪽 열은 발주 문서가 사업자의 전문성에 맡긴 판단입니다. 다만 기준을 사업자가 세우면 같은 시스템에 사업자마다 다른 유형과 견적이 나올 수 있고, 발주 기관은 그 차이를 검증할 자체 수치가 없습니다. 사전 진단은 사업 전에 기준과 수치를 함께 제시합니다. 이슈 한 건당 0·1·3·5·7·13점으로 매기는 Story Point 등급표와 필수 수정 여부 분류를 분석 도구 공식 문서의 정의대로 적용하고, 그 결과를 애플리케이션별 점수와 일수로 남깁니다. 이렇게 하면 발주 기관은 상세설계 사업자가 제시한 기준을 자체 수치와 대조할 수 있습니다.

7.1.3 컨테이너화가 어려운 대상의 처리 방침 부재

발주 문서의 예외 관련 서술

컨테이너로 옮기기 어려운 대상을 어떻게 처리할지를 정한 조항은 상세설계 제안요청서에 없습니다. 가장 가까운 서술은 두 가지인데, 하나는 Windows IIS나 PHP로 개발된 경우처럼 3계층 구성을 유지하기 어려운 경우를 예외로 둔다는 서술이고, 다른 하나는 사업기간 안에 끝낼 수 없는 전면 재개발을 제외한다는 조항입니다[S36]. 두 서술 모두 어떤 사유가 확인될 때 가상화(VM) 전환을 허용하는지, 그 판정을 누가 언제 내리는지는 다루지 않습니다.

본사업 단계에서는 이 판단이 조항 없이 구성 수량으로 나타납니다. 2025년 지방교육청 통합누리집 전환 본사업은 목표 구성을 클러스터 2식, 노드 20식, VM 17식으로 제시했습니다. 재활용하는 상용 소프트웨어 다섯 종은 VM이나 워커 노드에 배치하고, VOD, DNS, 접속대기 기능은 VM에 배치합니다[S41]. 각 VM이 어떤 사유로 컨테이너 전환에서 제외됐는지, 그 사유가 해소되면 언제 다시 검토하는지는 문서에 기록돼 있지 않습니다.

사전 진단의 예외 사유표와의 대응

예외 처리 기준이 없으면 VM 목록은 사업을 수행하는 도중에 정해집니다. 이 경우 일정이나 익숙함 같은 편의가 사유에 포함되기 쉽고, 한번 VM에 배치된 대상은 재검토 없이 남습니다. 컨테이너 전환이 기본이고 가상화 전환은 사유가 확인된 예외라는 원칙을 지키려면, 예외 판정이 본사업 전에 끝나 있어야 합니다. 6장의 예외 사유표가 이 판정 기준이며, 사전 진단은 대상마다 사유, 해소 조건, 재검토 시점을 기록한 목록을 산출물로 남깁니다.

발주 문서의 서술	6장 기준에 따른 사전 진단의 판정
Windows IIS로 개발된 경우의 예외	리눅스 전환을 먼저 검토하고, 어려우면 화면이 없는 서버 형에 한해 Windows 컨테이너를 검토하며, 두 단계를 통과하지 못할 때만 예외로 기록

발주 문서의 서술	6장 기준에 따른 사전 진단의 판정
PHP로 개발된 경우의 예외	개발 언어는 6.4.1항의 예외 사유에 들어 있지 않으므로, 리눅스에서 실행할 수 있게 만들 수 있으면 6.3절의 판정 기준과 단서 세 가지를 적용해 컨테이너 전환 대상으로 검토
전면 재개발 제외	재설계 등급 이슈의 점수와 건수를 제시해 재개발 범위에 대한 판단 근거 제공
재활용 상용 소프트웨어의 VM 배치	제작사의 컨테이너 지원 여부와 라이선스 조건을 확인한 뒤 사유와 재검토 시점 기록

출처: 상세설계·본사업 제안요청서의 서술과 컨테이너 공식 문서[S36][S41][S03][S04]

표의 오른쪽 열은 공식 문서에 적힌 문장이 아니며, 6.3절의 판정 기준과 6.4절의 예외 사유를 발주 문서의 서술에 적용한 결과입니다. 표의 왼쪽 열에 있는 대상이 곧바로 VM으로 가는 것은 아닙니다. 예를 들어 Windows 서버에서 실행되는 애플리케이션도 화면이 없는 서버형이면 Windows 컨테이너로 옮길 수 있고, Kubernetes는 Windows 워커 노드를 지원합니다. 따라서 발주 문서가 「예외」라고 적은 대상 가운데 일부는 진단을 거치면 컨테이너 전환 대상으로 분류됩니다. 진단 후에도 예외로 남는 대상은 사유가 기록된 목록으로 본사업 제안요청서에 실을 수 있습니다.

7.2 상세설계 대상 시스템에 반복해 나타나는 구버전 스택과 Java 외 언어

2024년과 2026년 상세설계 제안요청서에는 대상 시스템의 현행 스택이 실려 있습니다. 다섯 건의 문서를 함께 보면 구버전 Java, 상용 WAS, Unix 계열 OS가 차수마다 다시 나타나고, C·PHP·ASP·Python처럼 Java가 아닌 언어로 작성된 시스템도 반복해 포함됩니다[S36][S37][S38][S39][S40]. 따라서 5장에서 다룬 현대화 병행과 6장의 예외 판정은 실제 공공 전환 대상의 현황에서 나온 과제입니다.

7.2.1 Unix·상용 WAS·상용 DBMS로 구성된 2026년 상세설계 대상

2026년 대상 세 시스템의 구성

2026년 2차 상세설계 대상 세 시스템 가운데 두 시스템은 HP-UX, AIX, Solaris 계열 Unix를 주력 OS로 쓰고, 세 시스템 모두 상용 WAS와 상용 DBMS 위에서 운영됩니다. 2024년 1차 대상이 리눅스와 x86 서버 위주였던 것과 비교하면, 2026년 사업은 Unix와 구형 상용 소프트웨어의 비중이 높은 시스템을 대상으로 삼았습니다[S36][S37].

대상 시스템(업무 분야)	OS	WEB·WAS	DBMS	언어·프레임워크
보험 업무 시스템	HP-UX 11.31, AIX 7.1·5.3	IBM HTTP Server 6.1, WebSphere 6.1	Altibase 5.3	Java, C 외
차량 검사 관리 시스템	SunOS 5.10, AIX 7.1, Windows Server, CentOS 7	WebtoB, JEUS	Sybase	Java, 전자정부 표준 프레임워크

대상 시스템(업무 분야)	OS	WEB·WAS	DBMS	언어·프레임워크
수출 정보 시스템	CentOS 7.7	WebtoB 4, JEUS 6	Oracle 11g	Java·JSP, 전자정부 표준프레임워크 3.0.5

출처: 2026년 2차 상세설계 사업 제안요청서의 대상 시스템 현황[S36]

C 언어와 JDK 버전 공백

이 표에서 사전 진단이 먼저 확인할 사항은 두 가지입니다. 하나는 보험 업무 시스템의 개발 언어에 C가 들어 있다는 점입니다. Unix 장비에서 빌드한 C 바이너리는 x86 리눅스에서 그대로 실행되지 않으므로 리눅스로 다시 빌드하고 검증해야 하며, 이 작업은 5장에서 다룬 네이티브 코드의 난이도 요인과 같습니다. NIA 가이드의 설계 기준 예시도 Unix에서 리눅스로 옮기는 C 언어 재컴파일을 난이도 「상」으로 분류합니다[S45].

다른 하나는 세 시스템 모두 JDK 버전이 문서에 없다는 점입니다. JVM이 컨테이너의 자원 한도를 기본으로 인식하는 것은 JDK 10부터이고 JDK 8 계열에서는 8u191부터이므로, 버전을 모르면 컨테이너 전환에 Java 상향이 함께 필요한지를 판단할 수 없습니다[S07]. 실제 버전과 업데이트 번호를 확인하는 조사가 현황분석의 첫 과업이 됩니다. 이 정보 공백은 1장에서 다룬 문제, 즉 제안요청서에 품명만 있고 버전이 없는 상황과 같습니다.

7.2.2 JDK 1.6~1.8과 C·PHP·ASP가 혼재하는 2024년 상세설계 대상

2024년 차수별 대상 스택

2024년 네 차수의 대상에는 Unix가 거의 없는 대신 Java 버전과 미들웨어가 넓게 흩어져 있습니다. JDK 1.6, 1.7, 1.8이 한 사업 안에 섞여 있고, Tomcat은 7.0과 8.5가, 전자정부 표준프레임워크는 1.2부터 3.9까지 쓰입니다. 아래 표는 차수별로 눈에 띄는 구성만 요약한 것입니다.

차수	Java·프레임워크	WEB·WAS·DBMS	Java 외 언어와 특이 사항
1차	JDK 1.7·1.8	JEUS 8, Tomcat 8, 오픈소스 기반 WAS, Tiberio 6, Oracle 19c	Windows Server 2012 포함, Unix 없음
2차	JDK 1.6·1.7·1.8	오픈소스 기반 WAS, Tomcat 8.5, JEUS, Oracle, Tiberio, Cubrid 9, PostgreSQL 11.8	해당 없음
3차	전자정부 표준프레임워크 3.9	WebLogic, WebtoB 5·JEUS 8, Tomcat 7.0, Oracle 11g·12c	AIX 위의 PHP 홈페이지 1건
4차	JDK 1.6~1.8, 전자정부 표준프레임워크 1.2	JEUS 6, IIS 10, Tiberio 5	ASP, PHP 5, Python

출처: 2024년 1~4차 상세설계 사업 제안요청서의 대상 시스템 현황[S37][S38][S39][S40]

구버전 Java와 Java 외 언어의 진단 범위

JDK 1.6과 1.7에는 컨테이너 자원 한도를 인식하는 기능이 전혀 없는데, 이 기능은 JDK 10에서 기본으로 켜졌고 JDK 8 계열에는 8u191에 반영됐기 때문입니다[S07]. 따라서 이 버전의 애플리케이션은 컨테이너로 옮기는 작업과 Java 상향을 분리할 수 없습니다. Tomcat 8.5도 2024년 3월 31일에 지원이 끝났으므로, 대상 시스템의 미들웨어도 전환과 함께 올려야 합니다[S15].

Java가 아닌 언어는 진단 방법이 달라지는데, 애플리케이션 현대화 분석 도구가 정식으로 지원하는 언어가 Java이기 때문입니다. CNAMT 문서에서 Python과 Node.js 분석은 기술 미리보기(Technology Preview) 단계이고 .NET 분석은 개발자 미리보기(Developer Preview) 단계이며, C·C++·COBOL은 지원 대상으로 나오지 않습니다. 그래서 PHP, ASP, C로 작성된 시스템은 정적 분석 점수 대신 체크리스트와 인터뷰, 수작업 검토로 진단해야 하고, 그만큼 진단 기간과 결과의 정밀도가 달라집니다.

같은 사업 안에서도 대상 시스템의 현행 수준은 크게 다릅니다. 4차 대상 가운데 한 기초자치단체 시스템은 이미 Docker, JDK 11, Spring Boot 2.7을 쓰고 있는 반면, 같은 차수의 다른 시스템은 OpenJDK 1.6과 전자정부 표준프레임워크 1.2, JEUS 6 위에 있습니다[S40]. 대상마다 전환 공수가 크게 다르므로, 시스템 수만으로 예산을 나누면 실제 작업량과 맞지 않습니다.

7.3 NIA 안내서와 가이드에 없는 정량 기준 일곱 가지

NIA의 2021년 「클라우드 네이티브 정보시스템 구축 발주자 안내서」와 2025년 「클라우드 네이티브 구축·운영 가이드」는 성숙도 모델, 도입 적합성 체크리스트, 전환유형 정의, 비용 산정의 틀을 제공합니다. 반면 대상별 전환 방식과 공수를 정하는 데 필요한 정량 기준 가운데 다음 일곱 가지는 두 자료에 없습니다.

- 성숙도 배점과 등급 환산식(가이드 본문에 없고 별첨 파일로 제공)
- 전환유형 선택과 MSA 적용 여부의 정량 기준
- 모놀리스(하나의 배포 단위로 구성된 애플리케이션)를 컨테이너로 전환한 상태에 해당하는 성숙도 단계(2025년 모델)
- 리눅스가 아닌 워크로드의 처리 방침과 VM으로 남기는 기준
- 컨테이너 전환용 난이도·공수 표
- 전자정부 표준프레임워크 버전과 컨테이너 베이스 이미지 기준
- 애플리케이션 코드 수준의 진단 방법

가이드를 따르는 것만으로는 대상 시스템마다 전환 방식과 공수를 정할 수 없습니다. 가이드도 체크리스트 결과가 최종 판정이 아니며 상세설계에서 정해진다는 한계를 본문에 적어 두었습니다[S43][S45].

7.3.1 도입 필요 여부를 묻고 전환 가능 여부와 비용은 묻지 않는 NIA 체크리스트

2021년 안내서의 10문항 체크리스트

2021년 발주자 안내서는 사업기획 단계에서 발주자가 직접 답하는 10문항 체크리스트를 제시합니다. 문항은 정책 변화 대응, 신기술 도입, 잦은 변경 요청, 장애와 접속 지연 경험, 기능과 DB의 분리 가능성, 전문 인력과 자동화 도구 보유 여부 등을 묻습니다. 「예」가 5개이면 클라우드 네이티브 도입이 가능하고 6개 이상이면 적극 도입 대상으로 판정합니다[S43]. 그런데 이 문항 가운데 애플리케이션의 기술 스택, 버전, 코드 종속성을 확인하

는 문항은 없습니다. 10문항 체크리스트가 판정하는 것은 도입 필요성이며, 옮길 수 있는지와 비용이 얼마인지는 범위 밖입니다.

2025년 가이드의 현황 진단과 요구수준 평가

2025년 가이드는 체크리스트를 두 종류로 바꿨는데, 현황 진단은 예와 아니오로 답하는 15문항이고, 요구수준 평가는 5점 척도이며 전환 사업에는 32문항이 적용됩니다. 전환용 문항에는 적용 기술과 시스템 구조가 오래됐는지, 소프트웨어 다수가 오픈소스로 대체할 수 없는지를 묻는 문항이 새로 들어갔습니다. 다만 문항별 배점과 등급 환산식은 본문에 없고 별첨 파일로 남겨져 있습니다[S45].

가이드는 이 체크리스트의 한계도 함께 밝히고 있어, 체크리스트가 제시하는 도입 필요 단계는 최종 결과가 아니며 상세설계를 거쳐야 정해진다는 취지입니다. 2021년 안내서는 검토 주체를 단계별로 나눠, 사업기획 단계에서는 발주자가 체크리스트로 검토하고, 정보화전략계획 단계에서는 수행사가 상세 분석을 하며, 구축 단계에서 서비스별로 최종 결정합니다[S43]. 그 사이에서 대상이 지정되기 전에 코드와 구성을 근거로 전환 가능성과 공수를 재는 절차는 어느 단계에도 배정돼 있지 않습니다.

항목	NIA 체크리스트(2021년·2025년)	사전 진단
답하는 질문	도입할 필요가 있는가, 어느 수준까지 필요한가	컨테이너로 옮길 수 있는가, 기간과 공수는 얼마인가
입력	발주자의 예·아니오, 5점 척도 응답	13개 분류 체크리스트, 인터뷰, 소스·바이너리 정적 분석
보는 대상	조직의 요구, 운영 경험, 기술 현황에 대한 인식	기술 스택과 버전, 코드 종속성, 구성과 연계
결과	도입 필요 단계	애플리케이션별 점수, 일수, 전환유형, 예외 사유

출처: NIA 발주자 안내서와 구축·운영 가이드[S43][S45]

두 도구는 서로를 대체하지 않으며, NIA 체크리스트로 도입 필요성을 확인한 시스템에 사전 진단을 적용하면, 필요성과 실현 가능성을 순서대로 검토하게 됩니다. 반대로 사전 진단만으로는 그 시스템에 클라우드 네이티브 전환이 정책상 필요한지를 답할 수 없습니다.

7.3.2 IaaS 전환용으로만 제시된 난이도·공수 예시표

NIA 설계기준 예시의 구성

2025년 가이드의 부록에는 설계기준 예시가 실려 있고, 형식은 작업 내용, 전환 난이도, 서버당 M/D, 투입 직무입니다. 이 표는 IaaS 전환을 기준으로 작성했습니다. OS 항목에서는 Unix에서 리눅스로의 전환이 난이도 「상」이고, 애플리케이션 항목에서는 JDK 버전 변경에 따른 프레임워크 버전업이 「상」, 17 M/D로 공수가 가장 큰 항목에 속합니다. 아래 표는 애플리케이션과 직접 관련된 행만 발췌하고, 같은 작업을 사전 진단이 무엇으로 재는지를 나란히 적은 것입니다.

NIA 예시표의 작업(iaaS 전환)	난이도	서버당 M/D	사전 진단이 재는 방식
국산·공개 WEB·WAS 적용	중	9	상용 WAS 종속 코드의 필수 수정 이슈 건수와 점수
C 언어 재컴파일(Unix → Linux)	상	9	네이티브 라이브러리 목록 조사와 수작업 검토
하드코딩 수정	중	9	하드코딩된 IP와 경로의 발견 건수
런타임 버전 업그레이드 호환 확인	상	17	목표 Java 버전 기준의 이슈 건수와 점수
프레임워크 버전업(JDK 버전 변경에 따른 수정)	상	17	목표 Java·프레임워크 버전 기준의 이슈 건수와 점수
EJB 2.x·3.x와 WAS 라이선스 재활용	상	9	EJB 수와 유형, 관련 이슈 점수

출처: NIA 구축·운영 가이드 부록의 설계기준 예시와 분석 도구 공식 문서[S45]

컨테이너 전환용 대응표의 부재

정부 측 자료에서 전환 작업의 공수를 수치로 제시한 것은 이 표가 유일합니다. 하지만 이 표는 IaaS 전환용이어서 컨테이너 전환에서 새로 생기는 작업, 예를 들어 세션과 파일 같은 상태의 외부화, 설정의 환경 변수화, 컨테이너 이미지 구성은 다루지 않습니다. 가이드가 예시로 든 공개 소프트웨어 템플릿의 JDK는 OpenJDK 1.8로 적혀 있어, 목표 버전을 어디로 정할지에 대한 기준도 제공하지 않습니다[S45].

사전 진단에서는 설치·구성·테스트·협약에 해당하는 기본 작업 5일에 애플리케이션별 수정 일수를 더해 공수를 산정합니다. 진단 사례에서 애플리케이션 한 개당 산정 일수는 5일에서 27일 범위였습니다(4장). 다만 NIA 표의 단위는 서버당 M/D이고 사전 진단의 단위는 애플리케이션당 일수이므로, 두 수치를 그대로 더하거나 환산해서는 안 됩니다. 두 수치는 같은 작업을 서로 다른 기준으로 본 값으로 나란히 놓고, 차이가 큰 항목의 원인을 확인하는 용도로 씁니다.

7.3.3 「전환 불가」 예시 가운데 점수와 공수로 바꿔 제시할 수 있는 항목

가이드의 전환 불가 판정 예시

2025년 가이드의 설계기준 예시는 전환유형에 「전환 불가(Retain)」를 두고, 그 판정 예시로 여덟 가지를 듭니다. 소프트웨어 제작업체가 없거나 특정 소프트웨어에 종속된 경우, 상용 어플라이언스를 쓰는 경우, 현재 버전의 소스코드가 없는 경우가 여기에 들어 있습니다. 그런데 같은 목록에 애플리케이션의 WAS 라이브러리 종속 수정, EJB 2.x로 구현됐고 WAS가 지원 종료 대상인 경우도 함께 들어 있습니다.

뒤의 두 항목은 정적 분석으로 계량되는 항목입니다. 분석 도구는 대상 환경이 지원하지 않는 벤더 전용 인터페이스 사용을 필수 수정(Mandatory) 이슈로 분류하고, 이슈마다 Story Point를 매겨 발견 횟수와 곱한 총점을

내며, 보고서에는 EJB 목록이 유형별로 실립니다. 따라서 사전 진단은 이런 항목을 「불가」로 분류하지 않고 점수와 수정 일수로 제시합니다.

목록	가이드의 전환 불가 예시	사전 진단의 처리
계량 가능	WAS 라이브러리 종속 수정, EJB 2.x 구현과 WAS 지원 종료	정적 분석 점수와 수정 일수로 제시
예외 사유 해당	제작업체 부재, 특정 소프트웨어 종속과 대체 불가, 상용 어플라이언스, 소스코드 미보유	6장의 예외 사유표로 판정하고 사유와 재검토 시점 기록
예산·추진 방식 판단	프로그래밍 언어 변경 필요, 연계 시스템 간 망 단절, 전환 비용이 초기 구축 비용의 50% 초과	재개발·단계 분할·폐기 가운데 추진 방식을 다시 정하는 근거로 제시

출처: NIA 구축·운영 가이드의 전환 불가 판정 예시[S45]

재분류에 따른 전환 범위 변화

세 목록은 뒤따르는 의사결정이 서로 다른데, 먼저 계량 가능 목록은 수정 후 전환할 수 있는 대상이므로 전환 범위에 다시 들어옵니다. 예외 사유 목록은 해소 경로를 먼저 시도하고, 해소되지 않을 때만 VM에 두되 사유를 기록합니다. 반면 비용 초과는 그 자체로 가상화 전환의 사유가 되지 않으며, 재개발이나 단계 분할처럼 추진 방식을 다시 정하는 근거로 다룹니다.

이 재분류는 예산에 직접 영향을 주는데, 「불가」로 분류된 시스템은 전환 범위에서 빠져 기존 환경에 남고, 지원이 끝난 OS와 미들웨어의 유지 비용이 계속 발생합니다. 그 가운데 일부가 「수정 후 전환 가능」으로 바뀌면 전환 범위와 총사업비가 함께 달라집니다. 의사결정권자는 사전 진단 결과에서 계량 가능 목록의 점수와 일수를 먼저 확인하고, 이 대상을 본사업 범위에 넣을지를 정하면 됩니다.

7.3.4 2021년 4단계와 2025년 6단계 성숙도 모델에서 달라지는 Level 3의 뜻

두 성숙도 모델의 단계 구성

NIA의 성숙도 모델은 2021년 안내서에서 4단계였고 2025년 가이드에서 6단계로 바뀌었습니다. 두 모델 모두 Level 1은 IaaS로 이전한 단계이며, 달라지는 부분은 컨테이너가 등장하는 위치입니다. 2021년 모델의 Level 2는 PaaS와 컨테이너를 쓰는 모놀리식 구성입니다. 반면 2025년 모델에서 컨테이너는 MSA, 배포 자동화, 데브옵스와 같은 Level 3에 있습니다.

단계	2021년 발주자 안내서	2025년 구축·운영 가이드
Level 0	기존 애플리케이션(온프레미스, 모놀리식)	기존 애플리케이션(Initial)
Level 1	클라우드 준비단계(IaaS, 모놀리식)	클라우드 시작 단계(Entry): IaaS 전환, VM 기반
Level 2	클라우드 친화단계(PaaS와 컨테이너, 모놀리식)	클라우드 활용 단계(Operate): PaaS 도입

단계	2021년 발주자 안내서	2025년 구축·운영 가이드
Level 3	클라우드 네이티브단계(MSA)	클라우드 네이티브 도입 단계 (Scale): MSA, CI/CD, 컨테이너, 데브옵스, 비핵심 시스템부터
Level 4	없음	클라우드 네이티브 최적화 단계 (Optimize)
Level 5	없음	클라우드 네이티브 혁신 단계 (Innovate)

출처: NIA 발주자 안내서와 구축·운영 가이드의 성숙도 모델[S43][S45]

같은 Level 3, 다른 범위 — 컨테이너가 등장하는 단계가 한 단계 올라갔습니다

NIA 성숙도 모델의 2021년 4단계와 2025년 6단계 비교

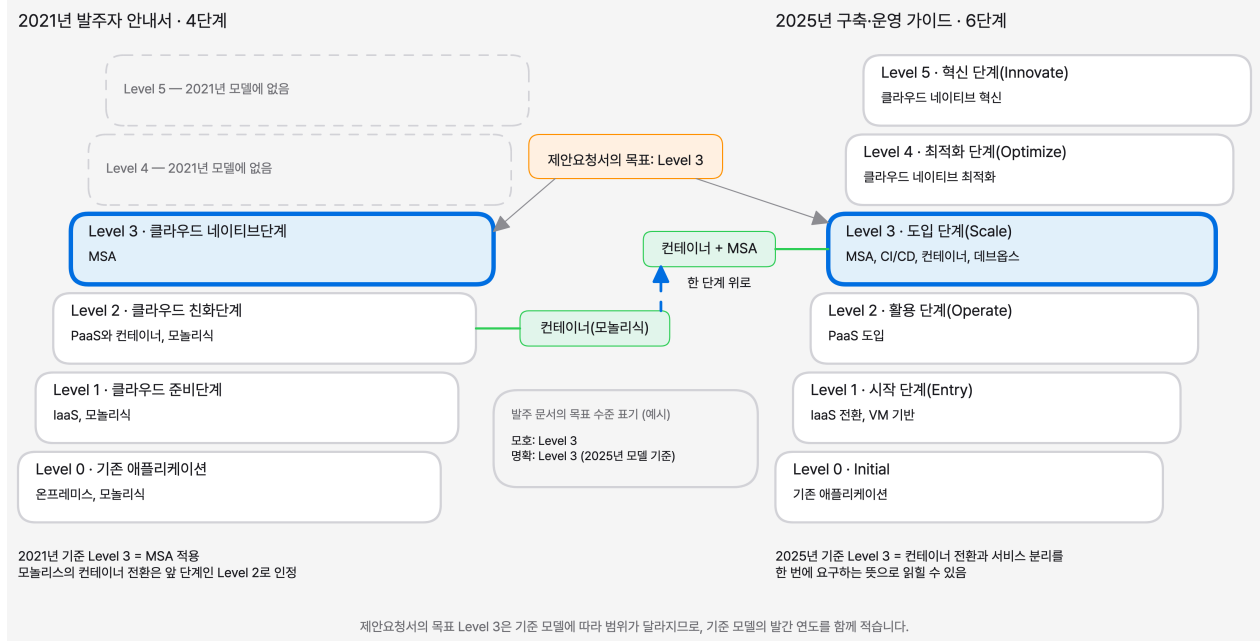


그림. NIA 2021년 4단계 모델과 2025년 6단계 모델에서 컨테이너가 등장하는 단계의 차이

발주 문서에 적는 목표 수준의 기준 모델

상세설계와 본사업 제안요청서는 전환 목표를 Level 3으로 적습니다[S36][S41]. 2021년 모델을 기준으로 하면 Level 3은 MSA 적용을 뜻하고, 모놀리식의 컨테이너 전환은 그 앞 단계인 Level 2로 인정됩니다. 그러나 2025년 모델에는 모놀리식을 컨테이너로 옮긴 상태에 해당하는 독립 단계가 없으므로, 같은 Level 3이라도 컨테이너 전환과 서비스 분리를 한 번에 요구하는 뜻으로 읽힐 수 있습니다. 여기에 더해 2024년 상세설계 제안요청서는 클라우드 준비, 친화, 심화, 네이티브로 이어지는 별도의 4단계 표현을 씁니다[S37].

목표 수준을 발주 문서에 적을 때는 기준 모델의 발간 연도를 함께 적어야 합니다. 기준 모델이 적혀 있지 않으면 사업자는 서비스 분리 범위를 서로 다르게 해석하고, 그 차이가 건적 차이로 이어집니다. 이 백서는 NIA 모델을 기준으로 삼고, 단계를 인용할 때 2021년 모델인지 2025년 모델인지를 밝힙니다. 컨테이너 전환을 먼저 하고

서비스 분리를 다음 단계로 두는 6장의 원칙은 NIA가 구성요소 도입 순서를 컨테이너, 데브옵스, MSA 순으로 제시한 것과 같은 방향입니다[S43].

7.4 본사업 제안요청서를 기준으로 사전 진단에서 미리 확정할 여덟 항목

본사업 제안요청서의 요구사항은 사업자가 수행 중에 충족해야 하는 조건이지만, 그 조건을 충족하는 데 필요한 정보는 대부분 사업 전에만 확보할 수 있습니다. 2025년 지방교육청 통합누리집 전환 본사업은 약 46억 원과 600일로 3개 시스템을 여섯 개 마이크로서비스로 전환하는 사업입니다[S41]. 이 문서의 요구사항과 계약특수 조건을 사업 전의 시각에서 다시 읽으면, 사전 진단에서 미리 정해 둘 항목 여덟 가지가 나옵니다.

사전 진단에서 미리 정할 항목	본사업 요구사항에서 대응하는 내용
소프트웨어 인벤토리와 버전, 언어, JDK	현행 소프트웨어가 품명으로만 제시된 상태에서 이기종 전환을 요구
상용·타사 소프트웨어별 처리 방침	재활용, 클라우드 제공 기능 대체, VM 잔류, 컨테이너 탑재 가운데 선택과 호환성 검증
서비스 분리 기준과 트래픽 최대치	접속이 몰리는 기능을 별도 서비스로 분리
전환 전 성능·운영 기준선	동등 수준 성능 보장과 성과 지표 비교
데이터 실태	DB 실태분석, 오류 데이터 정비, 이기종 DBMS 매핑
연계 전수	연계 31건의 방식, 방향, 주기
규모의 정확성	기능점수 재산정, 사업 중 신설분까지 포함한 전수 테스트
필요 자원 수량	VM, 노드, 스토리지 수량에 따른 전환 후 이용료

출처: 2025년 지방교육청 본사업 제안요청서와 전환지원 사업 계약특수조건[S41][S42]

의사결정권자는 이 표를 사전 진단의 범위를 정하는 기준으로 쓸 수 있습니다. 여덟 항목 가운데 본사업에서 변동 폭이 큰 항목은 소프트웨어 처리 방침, 성능 기준선, 자원 수량이므로 아래에서 차례로 설명합니다.

7.4.1 소프트웨어 인벤토리와 상용 소프트웨어별 처리 방침

버전까지 적은 소프트웨어 인벤토리

지방교육청 본사업 제안요청서의 현행 구성은 서버 38식과 소프트웨어 132식이고, Solaris 5식과 Windows 9식이 섞여 있습니다. 그런데 소프트웨어는 품명으로만 적혀 있어 버전, 개발 언어, JDK를 문서에서 확인할 수 없습니다. 목표 구성은 OS를 Rocky Linux로 단일화하고 국산·공개 소프트웨어 중심으로 바꾸는 이기종 전환입니다[S41]. 즉 사업자는 출발 버전을 모르는 상태에서 도착 버전까지의 수정 범위를 견적해야 합니다.

같은 기관의 2024년 상세설계 제안요청서에는 이 정보가 들어 있었고, JDK 1.6·1.7·1.8, ASP, PHP 5, WebtoB 4, JEUS 6·8, IIS 10, Oracle 11g·12c·19c, MySQL 5.1이 대상 현황으로 실려 있습니다[S40]. 따라서 상세설계에서 조사한 버전 정보가 본사업 문서로 전달되지 않으면 견적의 근거가 다시 사라집니다. 사전 진단의 인벤토리는 Java의 경우 메이저 버전뿐 아니라 업데이트 번호까지 기록하는데, 컨테이너 자원 한도 인식이 8u191에서, cgroup v2 인식이 8u372와 11.0.16에서 들어갔기 때문입니다[S07][S09].

상용 소프트웨어 처리 방침 네 가지

상용·타사 소프트웨어는 제품마다 네 가지 방침 가운데 하나를 정해야 합니다. 본사업 제안요청서에서 재활용 대상 다섯 종은 VM이나 워커 노드에 올리고, 상용 검색 제품은 클라우드가 제공하는 기능으로 대체합니다. WAS나 DBMS를 바꾸면 기존 상용 소프트웨어가 새 환경에서 동작하는지에 대한 호환성 검증이 따릅니다[S41].

- 재활용: 기존 라이선스를 새 환경으로 옮겨 사용
- 클라우드 제공 기능 대체: 제품을 제거하고 클라우드 서비스의 기능으로 교체
- VM 잔류: 6장의 예외 사유가 확인된 경우에 한해 사유와 재검토 시점을 기록하고 배치
- 컨테이너 탑재: 제작사가 컨테이너 지원판을 제공하거나 리눅스에서 실행되는 경우

처리 방침이 정해지지 않은 상용 소프트웨어는 본사업에서 가장 큰 변동 요인입니다. 제작사의 컨테이너 지원 여부, 라이선스가 하드웨어나 IP에 묶여 있는지, 대체 제품이 있는지는 계약 이후에 확인하면 일정과 비용에 바로 반영되기 때문입니다. 계약특수조건도 전환대상기관에 OS, WEB·WAS, DBMS의 전환 사항을 담은 체크리스트를 최종 확인하고 정보를 제공할 의무를 둡니다[S42]. 그러므로 제품별 방침을 사전 진단에서 정해 두면 발주 기관은 이 의무를 자체 자료로 이행할 수 있습니다.

7.4.2 동등 성능 보장을 증명하는 전환 전 성능·운영 기준선

본사업의 성능 보장 요구

본사업 제안요청서는 전환 전후에 기존 대비 동등 수준의 성능을 보장하도록 요구하고, 성과를 비교하는 기준은 개통 전 3년간 데이터의 평균으로 정해져 있습니다[S41]. 이 요구를 증명하려면 전환 전 시스템의 성능과 운영 지표가 수치로 남아 있어야 합니다. 운영 중인 시스템 가운데에는 응답시간이나 처리량을 지속해서 기록하지 않는 곳이 있고, 이 경우 「기존 대비 동등」을 판정할 기준값이 없습니다.

전환 전에만 수집할 수 있는 지표

기준선은 전환이 시작되면 다시 수집할 수 없는데, 기존 환경이 철거되거나 구성이 바뀐 뒤에는 같은 조건의 측정값을 얻을 수 없기 때문입니다. 따라서 사전 진단 단계에서 수집 항목과 수집 방법을 정하고, 측정 도구가 없는 시스템에는 애플리케이션 성능 모니터링 도구를 먼저 적용해 일정 기간의 값을 확보해야 합니다. 사전 진단의 체크리스트는 동시 사용자 수, 평균 응답시간, 빈번한 장애와 성능 이슈, 성능 모니터링 도구 사용 여부를 조사 항목으로 둡니다(3장).

기준선 항목	수집 내용	본사업에서의 용도
응답시간	주요 업무별 평균과 최대 응답시간	동등 수준 성능 보장의 판정 기준
처리량	동시 사용자 수, 시간대별 요청 건수와 최대치	서비스 분리 기준과 자원 산정의 입력
장애 건수	최근 3년 장애 건수와 중단 시간	성과 지표의 전환 전 평균
자원 사용률	서버별 CPU·메모리 평균과 최대 사용률	NIA 자원 산정식의 입력

기준선이 없으면 성능에 대한 이견이 생겼을 때 판정 근거가 없어, 발주 기관은 성능 저하를 입증하기 어렵고, 사업자는 동등 성능을 달성했음을 입증하기 어렵습니다. 반면 기준선이 문서로 남아 있으면 검수 조건을 수치로 적

을 수 있고, 성능 유지관리 기간의 성과 평가도 같은 값으로 진행할 수 있습니다.

7.4.3 VM·노드·스토리지 수량과 전환 후 이용료 예산

자원 수량과 운영 예산의 관계

전환 후에 필요한 VM, 노드, 스토리지 수량은 클라우드 이용료의 산정 근거가 됩니다. 구축비는 사업 기간에 한 번 집행되지만 이용료는 전환 후 매년 발생하고, 그 예산을 확보하는 책임은 전환대상기관에 있습니다[S41][S42]. 이 때문에 자원 수량이 과다하게 산정되면 그 차이는 구축 사업이 끝난 뒤에도 운영 예산에 계속 남습니다.

NIA 가이드의 자원 산정식

NIA의 2025년 가이드는 자원 산정식을 IaaS와 클라우드 네이티브로 나눠 제시합니다. CPU는 기존 코어 수에 평균 사용률과 여유율 1.3을 곱하고, Unix 장비는 보정값 0.8을 추가로 곱합니다. 클라우드 네이티브 구성에는 여기에 MSA 가중치 1.4가 더 적용되고, 메모리 여유율은 IaaS의 1.3보다 높은 1.8입니다.

항목	IaaS	클라우드 네이티브
CPU	기존 코어 × (Unix는 0.8) × 평균 사용률 × 1.3	같은 식 × MSA 가중치 1.4
메모리 여유율	1.3	1.8(× MSA 가중치 1.4)

출처: NIA 구축·운영 가이드의 자원 산정식[S45]

가상의 수치로 적용해 보면 입력값의 영향이 드러납니다. 예를 들어 16코어 x86 서버의 평균 사용률이 40%이면 IaaS 기준은 $16 \times 0.4 \times 1.3$ 으로 약 8.3코어이고, 클라우드 네이티브 기준은 여기에 1.4를 곱한 약 11.6코어입니다. 평균 사용률을 실제 측정값 대신 50%로 가정하면 결과는 약 14.6코어로 늘어납니다. 산정식의 계수는 고정돼 있으므로 결과를 좌우하는 것은 전환 전에 측정한 사용률의 정확도입니다.

예외로 남는 VM의 수도 이 예산에 직접 반영됩니다. VM은 게스트 OS를 각각 유지하므로 같은 애플리케이션을 컨테이너로 운영할 때보다 자원을 더 많이 차지하고, OS 패치와 라이선스 관리 대상도 그만큼 늘어납니다. 따라서 VM은 목표 비율로 정하는 대상이 될 수 없고, 예외 사유를 통과한 대상의 수로만 정해져야 합니다. 사전 진단이 VM 목록을 사유와 함께 미리 정해 두면 발주 기관은 전환 후 이용료를 근거 있는 수량으로 편성할 수 있습니다.

7.5 대상 지정 이전 단계를 맡는 사전 진단과 상세설계의 역할 분담

사전 진단, 상세설계, 본사업은 시작 시점과 정하는 항목이 서로 다릅니다. 사전 진단은 대상이 지정되기 전에 어느 시스템을 올릴지와 컨테이너 전환 공수의 범위를 제시합니다. 상세설계는 지정된 시스템의 목표모델, 이행계획, 기능점수 기반 예산을 정하고, 본사업은 그 설계에 따라 전환을 수행합니다[S36][S41]. 상세설계가 있는데도 사전 진단이 필요한 이유는 상세설계에 무엇을 올릴지를 정하는 판단이 상세설계의 과업 범위 밖에 있기 때문입니다.

7.5.1 시스템당 수억 원 규모 상세설계에 앞서는 대상 선정 판단

상세설계 한 건의 비용 규모

상세설계는 시스템당 수억 원이 드는 단계이며, 2024년 사업은 21개 시스템에 약 56억 원으로 시스템당 평균 약 2억 7천만 원이었습니다[S35]. NIA 가이드의 상세설계 수립비 산정 예시도 같은 수준으로, 컨설팅 업무량 26.28에 단가 10,010,006원을 곱해 263,062,958원이 나오고, 부가세를 포함하면 2억 8,900만 원입니다 [S45]. 대상 시스템을 하나 추가하거나 제외하는 결정은 수억 원 단위의 예산 결정입니다.

그런데 어느 시스템을 상세설계에 올릴지, 그 시스템이 컨테이너로 전환될 수 있는지, 난이도가 어느 정도인지는 상세설계가 시작되기 전에 답해야 하는 질문입니다. 이 질문에 쓸 정량 기준이 발주 문서와 NIA 자료에 없으므로, 전환이 어려운 시스템이 준비 없이 상세설계에 올라가거나, 비교적 쉽게 전환할 수 있는 시스템이 후순위로 미뤄질 수 있습니다.

세 단계가 정하는 항목

사전 진단은 상세설계보다 짧은 기간에 대상 선정, 전환유형 분류, 공수 범위를 먼저 제시합니다. 그 결과는 상세설계 대상을 고르는 근거가 되고, 상세설계와 본사업의 예산과 범위를 편성하는 입력값이 됩니다. 아래 도식은 세 단계가 각각 정하는 항목을 구분한 것입니다.

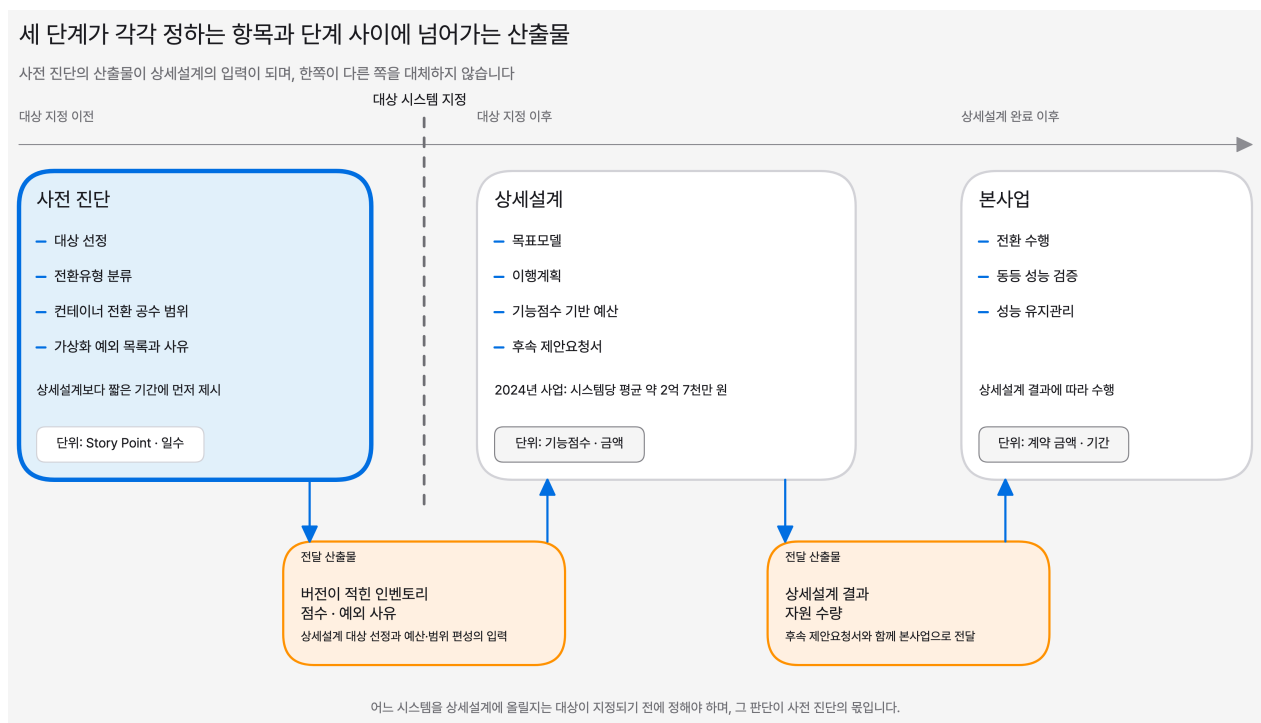


그림. 사전 진단·상세설계·본사업이 각각 정하는 항목과 단계 사이에 전달되는 산출물

단계	시작 조건	정하는 항목	단위
사전 진단	대상 지정 이전	대상 선정, 전환유형 분류, 공수 범위, 가상화 예외 목록	Story Point, 일수
상세설계	대상 지정 이후	목표모델, 이행계획, 소요 예산, 후속 제안요청서	기능점수, 금액
본사업	상세설계 완료 이후	전환 수행, 성능 검증, 유지관리	계약 금액, 기간

상세설계는 기능점수로 비용을 정하고 사전 진단은 Story Point와 일수로 난이도를 잹니다. 두 단위를 직접 환산하는 공식 계수는 없으므로, 사전 진단의 점수는 기능점수를 대신하지 않고 대가 산정 경로를 고르는 근거로 쓰입니다(4장). 두 단계는 사전 진단의 산출물이 상세설계의 입력이 되는 관계이며, 한쪽이 다른 쪽을 대체할 수 없습니다.

7.5.2 앞 단계 컨설팅 결과가 재산정된 사례와 진단 결과의 재사용 조건

2024년 상세설계에 포함된 재산출·재검토 과업

2024년 상세설계 사업에는 앞 단계 컨설팅의 결과를 다시 다루는 과업이 포함됐습니다. 4차 사업에는 기존에 컨설팅을 받은 4개 기관의 소요 예산을 검증하고 재산출하는 과업과 제안요청서 작성 지원이 포함돼 있고, 3차 사업은 2022년 심층 컨설팅 결과를 환경 변화를 고려해 재검토하도록 했습니다[S39][S40]. 두 과업 모두 앞 단계의 결과가 다음 단계에서 그대로 쓰이지 못한 사례입니다.

제안요청서는 재산정의 원인을 밝히지 않으므로 원인을 단정할 수는 없습니다. 다만 시간이 지나면 대상 시스템의 버전과 구성이 바뀌고, 산정 근거가 문서에 남아 있지 않으면 다음 사업의 사업자는 수치를 검증할 방법이 없어 다시 계산하게 됩니다. 규모 수치가 발주 단계에서도 일정하지 않은 예도 있는데, 2025년 본사업 제안요청서는 같은 문서 안에서 누리집 수를 130개와 133개로, 1,042개와 1,060개로 다르게 적고 있습니다[S41].

진단 결과가 다시 쓰이기 위한 조건

사전 진단 결과가 상세설계와 본사업에서 다시 산정되지 않으려면, 결론뿐 아니라 그 결론에 이르는 근거 자료가 함께 남아야 합니다. 상세설계나 본사업의 사업자가 같은 입력으로 같은 수치를 재현할 수 있으면 검증은 대조 작업으로 끝나고, 재산정은 변경된 부분에 한정됩니다. 아래 표는 발주 기관이 진단 결과를 인수할 때 확인할 조건입니다.

재사용 조건	확인할 내용
버전까지 적힌 인벤토리	OS, WAS, DBMS, 프레임워크, JDK의 버전과 업데이트 번호, 조사 기준일
점수의 산정 근거	분석에 쓴 도구 버전과 목표 환경, 이슈 분류별 건수와 Story Point
규모 자료	JSP·서블릿·클래스 수, EJB 수와 유형, 연계 건수
일수 환산 기준	기본 작업 일수, 점수의 시간 환산값, 월 20일 기준의 M/M 환산
기록된 예외 사유	가상화 예외 대상별 사유, 해소 조건, 재검토 시점
범위에서 제외한 항목	제외한 이슈와 사유(예: 목표 Java 버전에 따라 제외한 항목)

이 조건 가운데 조사 기준일과 목표 환경은 특히 중요합니다. 목표 Java 버전을 8로 두는지 17 이상으로 두는지에 따라 같은 애플리케이션의 점수가 크게 달라지므로, 목표가 적혀 있지 않은 점수는 상세설계 단계에서 해석할 수 없습니다(4장). 조사 기준일이 있으면 그 뒤에 바뀐 구성만 다시 조사하면 되므로 재조사 범위도 줄어듭니다. 정책결정권자는 사전 진단을 발주할 때 이 여섯 조건을 산출물 요건으로 넣어, 진단 결과가 상세설계 제안요청서의 대상 현황과 본사업 제안요청서의 구성 수량으로 이어지게 할 수 있습니다.

8장. 진단 이후: 우선순위, 파일럿, 단계적 확산

7장까지 다룬 사전 진단의 결과물은 애플리케이션별 Story Point, Mandatory 이슈 목록, 전환유형 분류, 가상화 전환 예외 목록입니다. 이 결과물은 클라우드 네이티브 전환 사업에서 어떤 대상을 먼저 옮기고 본사업을 어떻게 나눌지 정하는 입력으로 쓰입니다. 정부의 상세설계 사업은 대상 시스템이 지정된 뒤에 시작하며, 어느 시스템을 먼저 옮길지 정하는 기준은 발주 문서에 실려 있지 않습니다[S36]. 따라서 진단 결과를 전환 우선순위, 첫 전환 대상, 본사업의 단계 구성으로 옮기는 일은 발주 기관이 직접 해야 합니다. 아래에서는 그 세 가지를 정하는 기준을 차례로 제시한 뒤, 8.4절에서 상세설계·본사업을 발주하기 전에 사전 진단을 수행하기로 정하고 진단 대상 시스템과 범위를 고르는 방법으로 백서를 마무리합니다.

8.1 업무 중요도와 Mandatory 점수로 나누는 전환 우선순위 4분면

전환 우선순위는 업무 중요도와 Mandatory 점수 합이라는 두 값으로 정합니다. Mandatory는 애플리케이션 현대화 분석 도구가 고치지 않으면 빌드나 실행이 되지 않는다고 분류한 이슈이고, 점수 합은 발견 횟수에 건당 Story Point를 곱해 더한 값입니다. 업무 중요도는 기관이 정하는 값이고 Mandatory 점수는 진단에서 나오는 값이므로, 두 값을 교차시키면 중요하면서 수정이 적은 대상과 덜 중요하면서 수정이 많은 대상이 구분됩니다. 이 순서는 진단 수치에서 나온 것이어서, 예산 심의나 감사에서 순서의 근거를 설명할 때 진단 보고서를 그대로 제시할 수 있습니다.

8.1.1 빠른 성과·대형 과제·틈새·보류 및 폐기 네 구역의 처리 방침

4분면 구역별 처리 방침

4분면의 가로축은 Mandatory 점수 합이고 세로축은 업무 중요도입니다. 각 구역에는 처리 방침이 하나씩 대응하므로, 아래 표에서는 구역 이름보다 그 구역의 대상을 어떻게 처리하는지를 먼저 확인합니다.

처리 방침	업무 중요도	Mandatory 점수	구역 이름
가장 먼저 컨테이너로 전환하고, 첫 전환 대상도 이 구역에서 고릅니다	높음	낮음	빠른 성과
개념 검증(PoC, Proof of Concept)으로 위험을 먼저 확인한 뒤 단계를 나눠 전환합니다	높음	높음	대형 과제
표준 변환 절차로 묶어 일괄 전환합니다	낮음	낮음	틈새
전환 범위에서 빼고 폐기, 통합, 현행 유지 가운데 하나를 정합니다	낮음	높음	보류 및 폐기

전환 우선순위 4분면 — 중요하고 수정이 적은 대상부터 컨테이너로 전환합니다

업무 중요도와 Mandatory 점수 합을 교차시켜 구역마다 처리 방침을 하나씩 정합니다

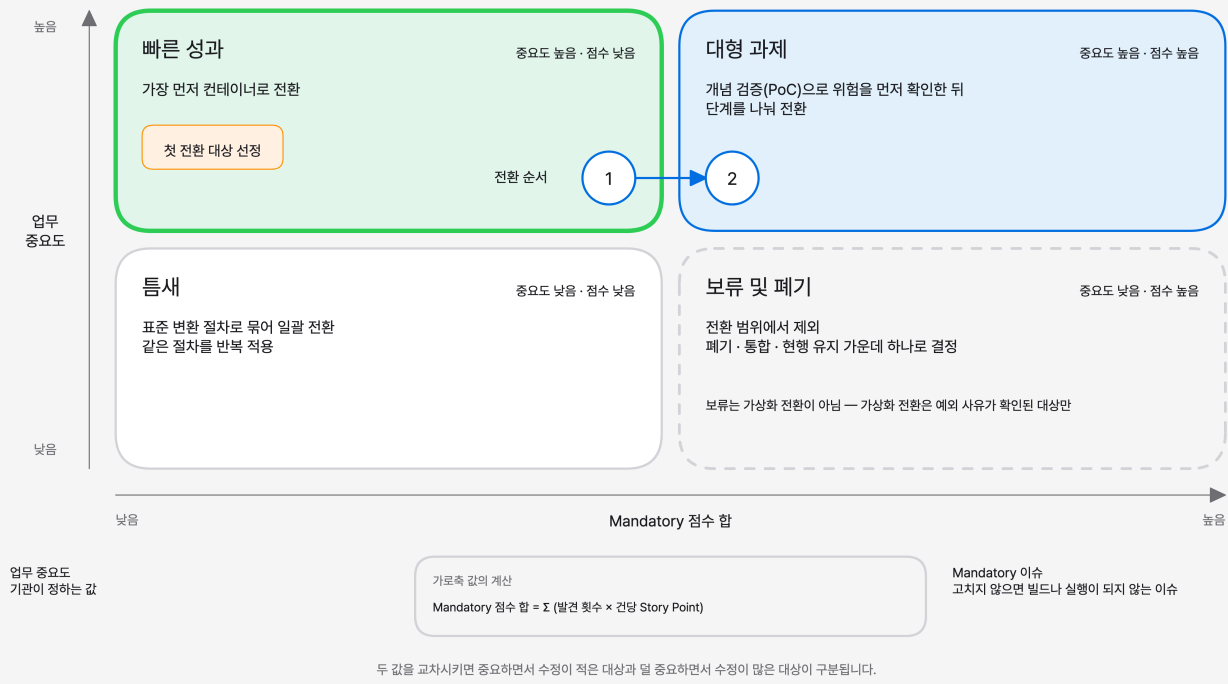


그림. 업무 중요도와 Mandatory 점수 합으로 나눈 전환 우선순위 4분면

보류·폐기 구역의 선별 효과

보류 및 폐기 구역의 대상을 먼저 가려내면 전환 범위와 예산이 함께 줄어듭니다. 업무 중요도가 낮는데 Mandatory 점수가 높은 대상은 수정 공수가 크고 전환으로 얻는 업무상 이득은 작기 때문입니다. AWS의 7R 분류에도 폐기(Retire)와 유지(Retain)가 전환 방식의 하나로 들어 있고[S33], NIA의 「클라우드 네이티브 구축·운영 가이드」(2025)도 설계기준 예시에 전환 불가(Retain) 유형을 둡니다[S45]. 다만 보류는 가상화 전환으로 옮긴다는 뜻이 아니며, 가상화 전환은 6장의 예외 사유가 확인된 대상에만 허용합니다. 비용이나 난이도가 높다는 사정은 재개발, 단계 분할, 폐기 가운데 전환 방식을 다시 정할 사유로 다룹니다.

Mandatory 점수의 해석 범위

4분면의 가로축에 Mandatory 점수만 쓰는 이유는 이 분류가 전환의 성립 여부와 직접 연결되기 때문입니다. Optional 이슈는 고치지 않아도 동작하므로 전환 직후의 일정으로 미룰 수 있고, Potential 이슈는 필수 여부를 판단할 정보가 부족해 검토가 더 필요한 항목입니다. Story Point는 작업 사이의 상대 크기를 일관되게 나타내는 값이며 인시로 반드시 환산되는 값은 아닙니다. 그래서 4분면은 대상 사이의 순서를 정하는 데 쓰고, 공수와 사업비는 4장의 절차에 따라 별도로 산정합니다.

8.1.2 상위 5개 모듈에 점수 83%가 집중된 사례와 표준 변환 대상 분리

Unix 전환 계획의 점수 분포

Unix 기반 시스템을 컨테이너로 전환하는 한 계획(기관명 비공개)에서는 상위 5개 모듈이 전체 점수의 83%를 차지했습니다. 이 계획은 점수와 외부 의존성을 기준으로 모듈을 나누었고, 점수가 높은 5건은 개념 검증을 먼저

수행하며 점수가 낮은 7건은 표준 변환 절차로 일괄 처리하도록 순서를 정했습니다. 위험이 큰 대상에는 검증 기간을 먼저 배정하고, 위험이 작은 대상에는 같은 절차를 반복 적용해 처리 기간을 줄인 구성입니다.

점수 분포에 따른 본사업 구성

이 수치는 한 사례의 분포이며, 다른 시스템에서 같은 비율이 나온다는 근거는 없습니다. 다만 점수가 모듈별로 고르게 분포하는지 일부 모듈에 집중되는지는 어느 진단에서든 확인할 수 있고, 그 결과에 따라 본사업의 구성이 달라집니다. 예를 들어 점수가 일부 모듈에 집중돼 있으면 그 모듈만 개념 검증 대상으로 분리해 나머지 모듈의 일정을 보호할 수 있습니다. 반면 점수가 고르게 분포하면 표준 변환 절차를 먼저 정해 전체 모듈에 적용하는 편이 효율적입니다. 진단 보고서를 받을 때는 총점과 함께 모듈별·애플리케이션별 점수 분포를 요구해야 합니다.

8.2 상태가 없고 기동이 빠른 서비스를 고르는 파일럿 선정 기준

파일럿(pilot, 시범 전환)은 본사업에 앞서 소수의 서비스를 실제로 컨테이너로 옮겨 운영해 보는 단계입니다. 파일럿의 목적은 가장 어려운 문제의 해결보다 전환 절차와 운영 방식의 확립에 있습니다. 이미지 빌드, 배포, 로그 수집, 모니터링 절차는 첫 대상에서 한 번 정하면 이후 대상에 반복 적용되므로, 첫 대상의 선정은 이후 전환 공수에 영향을 줍니다. 그래서 첫 대상은 4분면의 빠른 성과 구역 안에서 아래의 기술 조건을 충족하는 대상으로 고릅니다.

8.2.1 첫 전환 대상으로 적합한 서비스의 기술 조건

파일럿 대상 선정 점검표

첫 전환 대상의 조건은 기술 조건 네 가지와 진단 결과 두 가지를 합한 여섯 항목이며, 모두 진단의 체크리스트와 정적 분석 보고서로 확인할 수 있습니다. 아래 점검표에서 여섯 항목을 모두 충족하면 첫 대상의 후보가 됩니다.

점검 항목	적합 조건	확인 근거
상태 저장	세션과 파일을 로컬 디스크나 WAS 메모리에 두지 않음	체크리스트의 WAS 상세·WAS 종속성 분류
기동 시간	기동이 빠르고 기동 순서에 수작업이 없음	운영 담당자 인터뷰
외부 연계	연계 시스템 수가 적음	체크리스트의 연계 시스템 분류, 정적 분석의 하드코딩 IP 목록
소스와 빌드	소스와 빌드 절차를 기관이 보유함	체크리스트의 프레임워크 분류(소스 보유 여부)
Mandatory 점수	낮음(빠른 성과 구역)	정적 분석 보고서의 분류별 집계
예외 사유	6장의 가상화 전환 예외 사유에 해당하지 않음	예외 목록

점검표의 상태 저장과 기동 시간 조건은 12 Factor App이라는 애플리케이션 설계 원칙이 무상태 프로세스, 빠른 기동과 정상 종료라는 이름으로 다루는 항목입니다. 열두 원칙을 Kubernetes 설정 항목에 대응시켜 설명한 글은 부록의 함께 읽을 자료 [R3]에 실었습니다.

컨테이너 적합 대상의 공개 기준

Google Cloud의 Migrate to Containers 계획 가이드는 웹·애플리케이션 서버, Tomcat 같은 미들웨어, 중 소형 DB를 컨테이너 적합 대상으로 듭니다[S05]. 같은 가이드는 특수 커널 드라이버를 쓰는 VM, 특정 하드웨어에 의존하는 워크로드, 하드웨어 ID에 묶인 라이선스, 고성능·대용량 인메모리 DB, 데스크톱 애플리케이션을 부적합 대상으로 분류합니다[S05]. 따라서 Tomcat 위에서 동작하는 웹 애플리케이션처럼 적합 대상에 속하면서 위 점검표의 여섯 항목을 충족하면 첫 전환 대상의 후보로 올립니다. 로컬 디스크에 상태를 저장하거나 기동에 수십 분이 걸리는 대상도 컨테이너로 전환할 수 있지만 수정이 필요하므로 두 번째 이후 순서에 듭니다.

첫 대상에서 정하는 운영 절차

첫 대상에서 정해야 하는 절차는 이미지 빌드, 배포, 로그 수집, 모니터링 네 가지이며, 기존 이미지에 넣을 Java 런타임의 버전도 이때 함께 정합니다. JVM이 컨테이너의 자원 한도를 기본으로 인식하는 버전은 JDK 10과 JDK 8u191 이후이므로[S07], 파일럿에서 기존 이미지의 JDK 버전을 정해 두면 이후 대상은 같은 이미지를 재 사용할 수 있습니다. 절차와 기존 이미지가 먼저 정해지면 뒤따르는 대상의 공수는 애플리케이션별 수정 작업으로 한정됩니다.

8.2.2 고난도 대상을 첫 순서에 둘 때 약해지는 조직 내 지지

고난도 대상 우선 착수의 위험

가장 어려운 대상을 첫 순서에 두면 첫 성과가 나오는 시기가 늦어집니다. 대형 과제 구역의 대상은 개념 검증과 재설계를 거쳐야 하므로, 전환 절차와 운영 방식이 정해지기 전에 일정과 예산이 먼저 소진됩니다. 그 기간에는 조직이 확인할 수 있는 운영 결과가 없어서 전환 사업에 대한 내부의 지지가 약해집니다. 반면 빠른 성과 구역의 대상으로 시작하면 컨테이너 운영 실적이 먼저 확보되고, 그 결과가 다음 단계의 예산과 일정을 확보하는 근거가 됩니다.

내부 보고용 대상 선정 사유

첫 전환 대상의 선정 사유는 「업무 중요도가 높고 Mandatory 점수가 낮으며 가상화 전환 예외 사유에 해당하지 않는 서비스」로 적을 수 있습니다. 이 문장은 4분면의 구역과 점검표의 항목을 그대로 가리키므로, 보고를 받는 쪽이 선정 근거를 진단 보고서의 수치로 다시 확인할 수 있습니다. 대형 과제 구역의 대상은 전환 범위에 그대로 남아 있으며 개념 검증을 먼저 수행하는 순서에 배치됐다는 점을 함께 밝혀야 합니다. 그래야 어려운 대상을 뒤로 미룬다는 오해 없이 전환 순서를 설명할 수 있습니다.

8.3 컨테이너 전환을 먼저 완료하고 MSA를 뒤에 두는 본사업 분할

본사업은 컨테이너 전환과 서비스 분리를 한 번에 발주하는 방식과 단계로 나누어 발주하는 방식 가운데 하나로 구성합니다. MSA는 하나의 애플리케이션을 독립적으로 배포하는 여러 서비스로 나누는 구조이며, 기존 시스템을 컨테이너로 옮기는 일과는 별개의 결정입니다. NIA의 「클라우드 네이티브 정보시스템 구축 발주자 안내서」(2021)도 구성요소의 도입 순서를 컨테이너, 데브옵스와 배포 자동화, MSA 순으로 제시합니다[S43]. 따라서 본사업의 첫 단계는 컨테이너 전환의 완료로 한정하고, 서비스 분리는 그 뒤 단계에서 대상별로 결정합니다. 단계를 나누는 근거는 진단에서 나온 난이도 분포입니다.

진단, 컨테이너화, 배포 자동화, 오케스트레이션, 관측성의 순서로 단계마다 점검할 항목과 흔한 실패 요인을 정리한 글은 부록의 함께 읽을 자료 [R2]에 실었습니다.

8.3.1 이전과 서비스 분리를 별개 결정으로 나누는 단계 구성

본사업 첫 단계의 목표 범위

첫 단계의 목표는 기존 시스템을 컨테이너 기반으로 안전하게 옮기는 것입니다. 컨테이너 전환은 AWS와 Microsoft의 분류에서 모두 Replatform(플랫폼전환)에 해당하며[S33][S34], 정부 제안요청서가 정의한 플랫폼전환 유형에 가깝습니다[S36]. 반면 구조를 다시 설계하는 Refactor(재구성)에 대해 AWS는 대규모 이전에서 권하지 않는다고 밝힙니다[S33]. 대상 시스템이 여러 개인 전환 사업에서는 컨테이너 전환을 먼저 끝내고, 구조 재설계는 그 결과를 확인한 뒤에 판단하는 순서가 공개 권고와 일치합니다.

전면 재개발 대상의 범위 누락

정부 제안요청서는 Refactor(재구성) 유형 가운데 전면 재개발(Rebuild)을 사업 기간 안에 완료할 수 없으면 제외한다고 정합니다[S36]. 즉 컨테이너 전환과 서비스 분리를 한 사업에 묶으면 기간 안에 끝낼 수 없는 재개발 대상이 범위에서 빠지고, 그 대상은 후속 계획 없이 남게 됩니다. 단계마다 범위를 한정하면 이 문제를 피할 수 있으며, 첫 단계에서 모든 대상의 컨테이너 전환을 완료하고 서비스 분리가 필요한 대상은 다음 단계의 범위로 미리 지정해 두는 구성이 그 예입니다. NIA 안내서도 적합성 점수가 낮아 MSA를 이루는 경우에 컨테이너는 부분 도입 대상으로 둡니다[S43].

난이도 분포로 정하는 단계 수

단계의 수는 진단의 난이도 분포에 따라 정하며 고정된 값을 쓰지 않습니다. Mandatory 점수가 낮은 대상이 대부분이면 컨테이너 전환을 한 단계로 끝낼 수 있고, 대형 과제 구역의 대상이 많으면 개념 검증 결과를 반영해 컨테이너 전환을 두 단계 이상으로 나눕니다. 서비스 분리 단계에 올릴 대상도 진단 뒤에 정하며, NIA 교육자료(2021)는 MSA 적용을 판단하는 조건으로 잦은 배포의 필요, 성능 민감도, 분산 트랜잭션의 필요, 비용 대비 이득, 조직의 개발문화를 둡니다[S44]. 이 조건에 해당하는 서비스만 분리 대상으로 지정하면 서비스 분리 단계의 범위도 근거와 함께 한정됩니다.

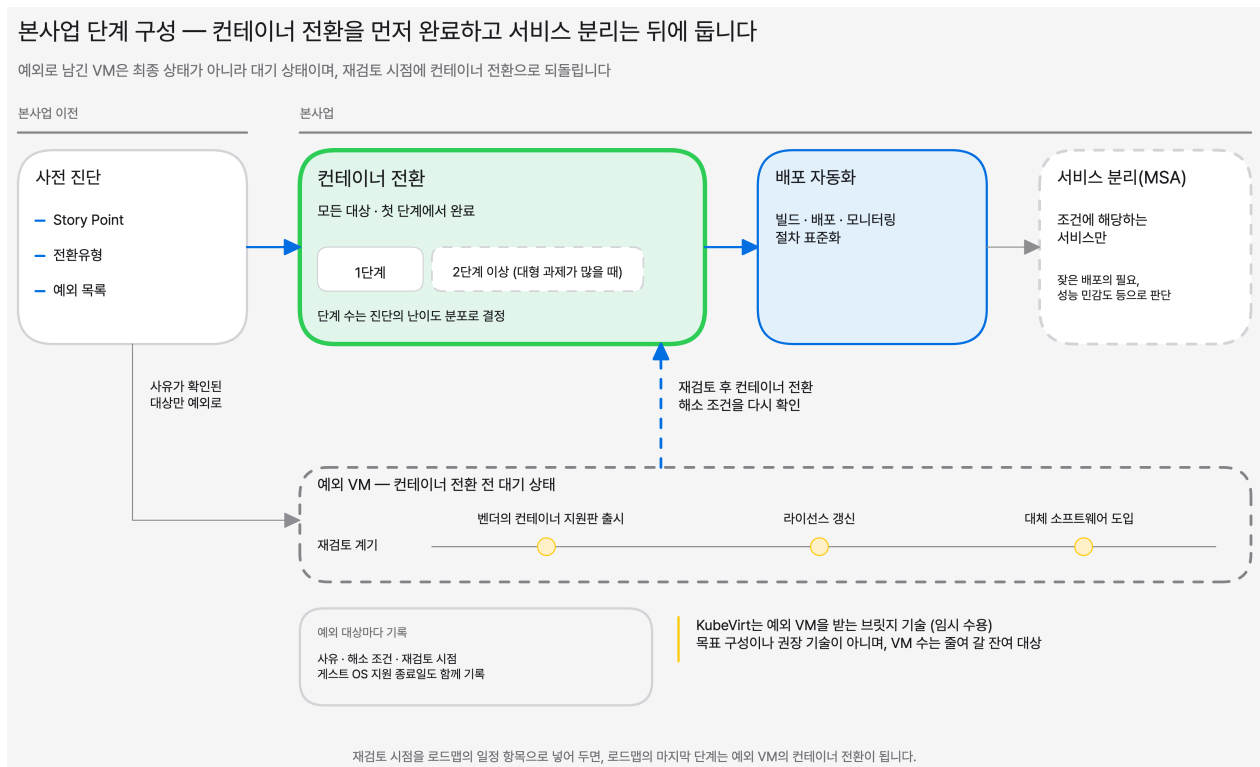


그림. 컨테이너 전환을 먼저 완료하고 서비스 분리를 뒤에 두는 본사업 단계 구성

8.3.2 예외로 남긴 VM의 재검토 시점을 로드맵에 넣는 절차

예외 기록의 로드맵 반영

가상화 전환 예외로 남긴 VM에는 6장의 절차에 따라 사유, 해소 조건, 재검토 시점을 기록합니다. 이 재검토 시점을 전환 로드맵의 일정 항목으로 넣어야 가상화 전환이 최종 상태로 굳지 않습니다. 정부 상세설계 사업의 과업에도 우선순위와 로드맵을 정하는 전환 이행계획이 들어 있으므로[S36], 예외 대상의 재검토 일정은 그 이행계획의 한 항목으로 반영할 수 있습니다.

예외 VM의 재검토 계기

재검토의 계기는 예외 사유가 달라지는 시기이며, 대표적인 예는 세 가지입니다. 아래 표는 계기별로 확인할 내용과 로드맵에 넣는 방법을 예시로 정리한 것입니다.

재검토 계기	확인할 내용	로드맵에 넣는 방법
벤더의 컨테이너 지원판 출시	지원 버전과 컨테이너 이미지 제공 여부	벤더의 출시 일정에 맞춘 정기 확인 항목
라이선스 갱신	하드웨어나 IP에 묶인 인증 방식의 변경 가능 여부	계약 갱신일 이전의 검토 항목
대체 소프트웨어 도입	대체 제품의 기능 범위와 호환성 검증 결과	대체 사업의 일정과 연계한 전환 항목

여기에 게스트 OS의 지원 종료일도 함께 적습니다. 게스트 OS의 지원이 끝나기 전에 재검토 일정이 있어야 하기 때문입니다.

브릿지 기술로 한정하는 KubeVirt

KubeVirt는 Kubernetes 위에서 VM을 실행하는 오픈소스 기술이며, 공식 소개도 대상을 쉽게 컨테이너화할 수 없는 기존 VM 기반 워크로드로 한정합니다[S06]. 6.5절에서 규정한 대로 KubeVirt는 예외 VM을 컨테이너로 옮기기 전까지 임시로 받는 브릿지 기술이며, 권장 기술이나 목표 구성이 될 수 없습니다. 한 지방교육청의 본사업 제안요청서는 컨테이너화가 어려운 대상을 처리하는 조항 없이 VM 17식에 배치합니다[S41]. 사전 진단에서 예외 목록과 재검토 시점을 먼저 정해 두면 이 공백이 채워지고, 로드맵의 마지막 단계는 예외 VM의 컨테이너 전환이 됩니다.

8.4 사전 진단 수행 결정과 진단 대상 시스템·범위 선정

상세설계와 본사업을 발주하기 전에 사전 진단을 먼저 수행하면, 컨테이너 전환 공수와 가상화 전환 예외 목록이 발주 문서에 반영된 상태로 사업이 시작됩니다. 2026년 상세설계 사업(2차)은 3개 시스템에 약 13억 3천만 원과 200일을 배정하지만, 대상 시스템은 사업 전에 이미 지정돼 있습니다[S36]. 어느 시스템을 올릴지, 그 시스템이 컨테이너로 갈 수 있는지는 그 앞 단계에서 정해야 하는 사항입니다. 따라서 발주 기관이 내릴 결정은 사전 진단의 수행 여부, 진단 대상 시스템, 진단 범위 세 가지입니다.

8.4.1 지원 종료 계층 수와 사업 일정으로 고르는 진단 대상 시스템

진단 대상 선정 점검표

진단 대상은 지원이 끝난 계층의 수, 발주 일정, 플랫폼의 종류로 고릅니다. 아래 점검표에서 해당 항목이 많은 시스템을 먼저 진단 대상에 올리며, 점검표는 기관 내부 검토에 그대로 쓸 수 있습니다.

점검 항목	확인 내용	먼저 진단하는 이유
지원 종료 계층	OS, Java, WAS, 프레임워크 가운데 지원이 끝난 계층의 수	계층마다 현대화 작업이 추가되어 공수 편차가 커짐
발주 일정	상세설계나 본사업의 발주 예정 여부와 시기	진단 결과를 발주 문서에 반영할 기간이 필요함
Unix 플랫폼	AIX, HP-UX, Solaris 위에서 운영하는지 여부	리눅스 전환 검증과 네이티브 코드 확인이 먼저 필요함
상용 WAS	JEUS, WebSphere, WebLogic 등의 사용 여부	대상 환경이 지원하지 않는 독점 API는 Mandatory 이슈로 분류됨
Java 외 언어	C, PHP, ASP 등의 포함 여부	분석 도구의 정식 지원 범위 밖이어서 수작업 진단이 필요함

지원 종료 계층의 확인 방법

지원 종료 여부는 벤더의 수명주기 안내로 확인합니다. 예를 들어 CentOS Linux 8은 2021년 12월 31일[S10], AIX 7.1 TL5는 2023년 4월 30일[S11], WebSphere Application Server 7과 8은 2018년 4월 30일[S17], Apache Tomcat 8.5는 2024년 3월 31일에 지원이 끝났습니다[S15]. OpenJDK 8·11의 지원 종료 시점은 배포판 공급자마다 다르므로 사용 중인 배포판의 수명주기 공지를 확인해야 합니다. 따라서 CentOS Linux 8과 Tomcat 8.5를 함께 쓰는 시스템은 두 계층에서 이미 지원이 끝난 상태이며, Java와 프레임워크의 버전까지 확인하면 해당 계층은 더 늘어날 수 있습니다.

발주 일정·플랫폼별 진단 순서

2026년 상세설계 사업(2차)의 대상 3개 시스템 가운데 2개는 Unix가 주력이고 3개 모두 상용 WAS와 상용 DBMS를 사용하며, JDK 버전은 제안요청서에 적혀 있지 않습니다[S36]. 이런 시스템은 버전 인벤토리부터 다시 조사해야 하므로 발주 전에 진단하는 효과가 큼니다. Unix 위의 Java 애플리케이션은 가상화 전환 예외에 해당하지 않고 리눅스 전환 검증 대상이므로, 진단에서 네이티브 라이브러리의 유무를 먼저 확인해야 컨테이너 전환 공수를 산정할 수 있습니다. 발주가 예정된 시스템은 발주 문서의 작성 일정보다 앞서 진단을 마쳐야 결과를 반영할 수 있습니다.

8.4.2 진단 범위에 포함할 애플리케이션 산출물과 반입 조건

진단 범위 결정 항목

진단 범위는 제출할 애플리케이션 산출물, 체크리스트, 운영 지표로 정하고, 반입·반출 절차와 분석 환경을 먼저 준비합니다. 아래 표는 범위를 정할 때 결정해야 하는 다섯 항목입니다.

결정 항목	정할 내용	유의 사항
애플리케이션 산출물	소스 또는 업무용 WAR·EAR 가운데 제출할 형태	타사 라이브러리 JAR만 제출하면 분석할 수 없음
체크리스트	13개 대분류의 작성 범위와 작성 담당자	애플리케이션 상세 (JSP·Servlet·EJB 수)는 공수 산정의 입력임
운영 지표	동시 사용자, 응답시간, 장애 이력의 수집 기간	전환 전 성능 기준으로 쓰임
반입·반출 절차	산출물과 분석 도구의 반입, 보고서의 반출 권한	보안 승인에 걸리는 기간을 일정에 반영함
분석 환경	별도 작업 PC와 OpenJDK 11 이상	기관 내부망에서 분석하면 사전 준비가 필요함

제출물 형태와 분석 가능 여부

제출물의 형태가 분석 가능 여부를 정하며, 대학 E의 진단에서 2건을 분석하지 못한 사유도 운영체제와 무관하게 제출물의 형태에 있었습니다. 제출된 파일이 타사 라이브러리 JAR이거나 업무용이 아닌 WAR였기 때문이며, 같은 진단에서 AIX 위의 Java 애플리케이션은 분석과 판정을 모두 마쳤습니다. 범위를 정할 때는 시스템별로 소스 보유 여부와 업무용 배포 파일의 위치를 먼저 확인합니다. 분석 도구가 정식으로 지원하는 언어는 Java이고 Python과 Node.js는 Technology Preview, .NET은 Developer Preview 단계이므로, Java 외 언어로 작성된 부분은 수작업 진단 범위로 따로 표시합니다.

버전 정보의 수집 수준

체크리스트의 버전 항목에는 메이저 버전과 함께 업데이트 번호까지 적어야 합니다. cgroup v2를 인식하는 JVM은 JDK 15, 11.0.16, 8u372 이후이기 때문에[S09], 같은 Java 11이라도 업데이트 번호에 따라 컨테이너 전환 때 필요한 조치가 달라집니다. 행정·공공기관 클라우드 네이티브 전환지원 사업의 계약특수조건은 전환대상기관에 OS, WEB/WAS, DBMS 전환 사항을 담은 체크리스트의 최종 확인 의무를 둡니다[S42]. 즉 체크리스트는 본사업 단계에서도 작성해야 하는 자료이며, 사전 진단은 그 작성 시기를 발주 전으로 앞당깁니다.

8.4.3 진단 결과를 상세설계·본사업 발주 문서에 반영하는 순서

진단 결과와 발주 문서의 대응

진단 결과는 네 가지 항목으로 발주 문서에 반영합니다. 상세설계 제안요청서의 과업은 현황분석, 목표모델 수립, 전환 이행계획, 전환 세부계획 네 건이므로[S36], 진단 결과의 각 항목은 아래 표와 같이 과업 하나씩에 대응합니다.

진단 결과 항목	상세설계 과업 범위	본사업 제안요청서
버전까지 적힌 인벤토리(OS, Java, WAS, 프레임워크, 언어)	현황분석의 조사 범위와 서식	현황 자료의 소프트웨어 구성 현황

진단 결과 항목	상세설계 과업 범위	본사업 제안요청서
전환유형 분류	목표모델의 전환 방식과 유형별 설계 기준	대상 시스템별 전환유형
가상화 전환 예외 목록과 사유	이행계획의 우선순위와 로드맵	VM 수량과 재활용 상용 소프트웨어의 배치
공수 범위(Story Point와 일수)	세부계획의 소요예산 산출 근거	사업 기간과 단계 구성

본사업 현황 자료의 버전 정보

한 지방교육청의 본사업 제안요청서는 서버 38식과 소프트웨어 132식의 현황을 싣고 있지만 버전, 언어, JDK는 빠져 있고 품명만 적혀 있습니다[S41]. 같은 문서는 제안요청서에 구체적으로 명시되지 않은 부분도 계약금액의 변동이 없는 범위에서 요구에 응하도록 정합니다[S41]. 그래서 현황 자료에 버전이 빠지면 그만큼의 작업이 사업자의 추정에 맡겨지고, 1장에서 본 범위와 비용의 불일치로 이어집니다. 진단에서 나온 인벤토리를 현황 자료에 실으면 제안사는 같은 정보를 바탕으로 공수를 산정할 수 있습니다.

공공 사업비의 산정 경로

공수 범위는 사업비 금액으로 바로 옮기지 않습니다. Story Point를 기능점수나 M/M으로 바꾸는 공식 계수는 없으며, 분석 도구의 공식 문서도 Story Point가 인시로 반드시 환산되지는 않는다고 밝힙니다. 공공 사업비는 「SW사업 대가산정 가이드」(2024년 개정판)의 경로로 따로 산정합니다. 이 가이드는 기능 변경량이 0으로 산정된 재개발에 대해 분석·설계 단계를 제외하고 구현·시험 단계의 개발비만 다시 산정하는 방식이 적절하다고 안내하며, 두 단계의 가중치는 32%와 25%입니다[S25]. 진단 결과는 대가산정 가이드의 경로 가운데 어느 것을 적용할지 고르는 근거로 씁니다(4장).

발주 전 결정 순서

발주 기관이 내릴 결정을 순서대로 적으면 다음 세 단계입니다.

1. 상세설계·본사업을 발주하기 전에 사전 진단을 먼저 수행하기로 정합니다.
2. 8.4.1항의 점검표로 진단 대상 시스템을 고르고, 8.4.2항의 결정 항목으로 제출 산출물과 진단 범위를 정합니다.
3. 진단 결과인 인벤토리, 전환유형 분류, 예외 목록과 사유, 공수 범위를 상세설계 과업 범위와 본사업 제안요청서의 현황 자료에 반영합니다.

전환의 기본은 컨테이너 전환이고 가상화 전환은 사유가 확인된 예외입니다. 사전 진단은 그 예외 목록과 컨테이너 전환 공수를 본사업 전에 정하는 절차이며, 위 세 단계 가운데 첫 번째 결정이 나머지 두 단계의 시작 조건입니다. 따라서 발주 기관이 먼저 작성할 문서는 진단 대상 시스템의 목록과 진단 범위의 초안입니다.

부록: 참고문헌과 용어 정의

이 부록은 본문에 인용한 참고문헌과 본문에 나오는 주요 용어의 정의를 모은 참조 자료입니다.

Appendix A — 참고문헌 (References)

본문에서 대괄호 번호로 인용한 출처입니다. 번호는 본문의 인용 번호와 같으며, 제안요청서와 공공 안내서처럼 공개 문서로 배포된 자료는 제목과 연도만 적었습니다.

- [S01] Docker. "What is a Container". <https://www.docker.com/resources/what-container/>
- [S02] Docker. "Docker Engine security". <https://docs.docker.com/engine/security/>
- [S03] Kubernetes. "Windows containers in Kubernetes". <https://kubernetes.io/docs/concepts/windows/intro/>
- [S04] Microsoft Learn. "Lift and shift to containers". <https://learn.microsoft.com/en-us/virtualization/windowscontainers/quick-start/lift-shift-to-containers>
- [S05] Google Cloud. "Migrate to Containers 계획 가이드". <https://docs.cloud.google.com/migrate/containers/docs/planning-best-practices>
- [S06] KubeVirt. "KubeVirt 프로젝트 소개". <https://kubevirt.io/>
- [S07] OpenJDK. "JDK-8146115: 컨테이너 감지와 자원 설정". <https://bugs.openjdk.org/browse/JDK-8146115>
- [S08] OpenJDK. "JDK-8170888: cgroup 메모리 한도 실험 지원". <https://bugs.openjdk.org/browse/JDK-8170888>
- [S09] OpenJDK. "JDK-8230305: Cgroups v2 컨테이너 인식". <https://bugs.openjdk.org/browse/JDK-8230305>
- [S10] CentOS. "CentOS Linux EOL". <https://www.centos.org/centos-linux-eol/>
- [S11] IBM. "AIX support lifecycle information". <https://www.ibm.com/support/pages/aix-support-lifecycle-information>
- [S12] Jakarta EE. "javax에서 jakarta 네임스페이스로". <https://jakarta.ee/blogs/javax-jakartaee-namespace-ecosystem-progress/>
- [S13] Spring. "Spring Boot 3.0 Release Notes". <https://github.com/spring-projects/spring-boot/wiki/Spring-Boot-3.0-Release-Notes>
- [S14] Spring. "Spring Framework Versions". <https://github.com/spring-projects/spring-framework/wiki/Spring-Framework-Versions>
- [S15] Apache Tomcat. "Which Version". <https://tomcat.apache.org/whichversion.html>
- [S16] 전자정부 표준프레임워크. "실행환경 저장소와 릴리스". <https://github.com/eGovFramework/egovframe-runtime>
- [S17] IBM. "WebSphere Application Server V7·V8 지원 종료". <https://www.ibm.com/support/pages/eos-ibm-websphere-application-server-v7->

and-v8

- [S18] endoflife.date. "Oracle JDK, CentOS, Spring Boot, Solaris 수명주기 집계".
<https://endoflife.date/>
- [S19] IBM developerWorks. (2005). "Guide to porting from Solaris to Linux on x86".
<https://public.dhe.ibm.com/software/dw/linux/l-solar/l-solar-pdf.pdf>
- [S20] Oracle. "Java Virtual Machine Specification, Java SE 21, 1장".
<https://docs.oracle.com/javase/specs/jvms/se21/html/jvms-1.html>
- [S21] Oracle. "JNI Specification (JDK 21)".
<https://docs.oracle.com/en/java/javase/21/docs/specs/jni/intro.html>
- [S22] OpenJDK. "JEP 400: UTF-8 by Default". <https://openjdk.org/jeps/400>
- [S23] IBM. "Semeru Runtimes Migration Guide".
<https://www.ibm.com/support/pages/node/6366719>
- [S24] 아이티데일리. "HPE, 신한금융투자 코어시스템 U2L 전환".
<https://www.itdaily.kr/news/articleView.html?idxno=88660>
- [S25] 한국소프트웨어산업협회. (2024). "SW사업 대가산정 가이드 2024년 개정판".
[https://data.cmpi.or.kr/program/board/\[2024.05.13\] 소프트웨어사업 대가산정 가이드.pdf](https://data.cmpi.or.kr/program/board/[2024.05.13] 소프트웨어사업 대가산정 가이드.pdf)
- [S26] 바이라인네트워크. (2025). "SW사업 대가산정 가이드 2025년 개정 보도".
<https://byline.network/2025/08/8112/>
- [S27] Commeyne, Abran, Djouab. (2016). "Effort Estimation with Story Points and COSMIC Function Points". <https://cosmic-sizing.org/publications/effort-estimation-story-points-cosmic-function-points-industry-case-study/>
- [S28] AWS Prescriptive Guidance. "Evaluating migration readiness".
<https://docs.aws.amazon.com/pdfs/prescriptive-guidance/latest/migration-readiness/migration-readiness.pdf>
- [S29] Microsoft. "Assess your workloads for cloud migration".
<https://learn.microsoft.com/en-us/azure/cloud-adoption-framework/plan/assess-workloads-for-cloud-migration> · "Azure Migrate application and code assessment for Java". <https://learn.microsoft.com/en-us/azure/migrate/appcat/java>
- [S30] Google Cloud. "RaMP overview". <https://docs.cloud.google.com/migration-center/docs/ramp-overview>
- [S31] 전자신문. (2021). "2021년 공공부문 클라우드 도입 컨설팅 보도".
<https://www.etnews.com/20210910000043>
- [S32] Help Net Security. (2011). "Gartner의 클라우드 이전 다섯 가지 방법".
<https://helpnetsecurity.com/?p=31394>
- [S33] AWS Prescriptive Guidance. "Migration strategies (7R)".
<https://docs.aws.amazon.com/prescriptive-guidance/latest/large-migration-guide/migration-strategies.html>
- [S34] Microsoft. "Select your cloud migration strategies".
<https://learn.microsoft.com/en-us/azure/cloud-adoption-framework/plan/select->

cloud-migration-strategy

- [S35] 디지털데일리. (2024). "2024년 클라우드 네이티브 전환 상세설계 사업 보도".
<https://www.ddaily.co.kr/page/view/2024041518243006840>
- [S36] 과학기술정보통신부·NIA. (2026). "2026년 클라우드 네이티브 전환 상세설계 사업(2차) 제안요청서".
- [S37] (2024). "2024년 클라우드 네이티브 전환 상세설계(1차) 사업 제안요청서".
- [S38] (2024). "2024년 클라우드 네이티브 전환 상세설계(2차) 사업 제안요청서".
- [S39] (2024). "2024년 클라우드 네이티브 전환 상세설계(3차) 사업 제안요청서".
- [S40] (2024). "2024년 클라우드 네이티브 전환 상세설계(4차) 사업 제안요청서".
- [S41] (2025). "2025년 클라우드 네이티브 전환 사업(지방교육청 통합누리집) 제안요청서".
- [S42] "행정·공공기관 클라우드 네이티브 전환지원 사업 계약특수조건".
- [S43] NIA. (2021). "클라우드 네이티브 정보시스템 구축 발주자 안내서".
- [S44] NIA. (2021). "클라우드 네이티브 교육자료 1~3차".
- [S45] NIA. (2025). "클라우드 네이티브 구축·운영 가이드".
- [S46] (2023). "공공부문 민간 클라우드 도입·활용을 위한 공공서비스 혁신 컨설팅 발표 자료".

함께 읽을 자료

아래 [R] 번호는 같은 주제를 다룬 공개 자료이며 본문 주장의 근거로 쓰지 않았습니다. 확인된 출처는 위의 [S] 번호로만 표시합니다.

- [R1] "클라우드 네이티브 전환 진단 발표자료". <https://www.msap.ai/msa/cloud-native-report/>
- [R2] (2026). "클라우드 네이티브 전환 로드맵 — 5단계로 그리는 마이그레이션 전략".
<https://www.cncf.co.kr/cloud-native/cloud-native-migration-roadmap/>
- [R3] (2026). "12 Factor App — 클라우드 네이티브 설계 원칙 12가지".
<https://www.cncf.co.kr/cloud-native/twelve-factor-app/>

Appendix B — 용어 정의 (Glossary)

본문에 나오는 용어를 한글 가나다순, 영문 알파벳순으로 정리했습니다.

용어	정의
가상화 예외 목록	컨테이너로 전환할 수 없음이 확인된 대상과 그 사유, 해소 조건, 재검토 시점을 적은 목록
게스트 OS	가상 머신 안에 설치되어 VM마다 따로 실행되는 운영체제
기능점수(Function Point)	사용자에게 제공되는 기능의 규모를 재는 단위. 공공 소프트웨어 사업 대가 산정의 기준

용어	정의
네임스페이스	Java 패키지 이름의 앞부분. Jakarta EE 9에서 javax가 jakarta로 바뀜
데브옵스	개발과 운영을 하나의 절차로 묶어 빌드와 배포를 자주, 자동으로 수행하는 방식
모놀리식	애플리케이션 전체를 하나의 배포 단위로 구성한 구조
배포 서술자	애플리케이션의 배포 설정을 적은 파일
본사업	상세설계 결과에 따라 전환을 실제로 수행하는 사업
사전 진단	본사업 발주 전에 대상 애플리케이션의 컨테이너 전환 가능 여부와 전환 공수를 수치로 확인하는 절차
상세설계	본사업에 앞서 대상 시스템의 목표모델, 이행계획, 소요예산을 정하는 선행 사업
애플리케이션 인벤토리	애플리케이션 목록과 OS, WAS, Java 버전, 연계 시스템 같은 구성 정보를 정리한 자료
애플리케이션 현대화 분석 도구	소스나 배포 파일을 규칙으로 검사해 전환 이슈와 Story Point를 내는 정적 분석 도구
엔디언	여러 바이트로 이루어진 값을 저장하는 바이트 순서
전자정부 표준프레임워크	공공 정보시스템 개발에 쓰는 Spring 기반 표준 개발 프레임워크
정적 분석	프로그램 실행 없이 소스 코드나 배포 파일을 규칙에 따라 검사하는 방법
컨테이너	애플리케이션과 실행 환경을 하나의 이미지로 묶어, 호스트 커널을 공유하는 격리된 프로세스로 실행하는 기술
파드(Pod)	Kubernetes가 컨테이너를 배치하고 관리하는 최소 단위
파일럿	본사업에 앞서 소수의 서비스를 실제로 컨테이너로 옮겨 운영해 보는 시범 전환
하이퍼바이저	물리 서버 위에서 여러 VM을 실행하는 가상화 소프트웨어
힙(heap)	JVM이 객체를 저장하는 메모리 영역
cgroup	리눅스 커널이 프로세스 묶음의 CPU와 메모리 사용량을 제한하는 기능. v1과 v2 두 구조가 있음
CI/CD	지속적 통합·배포. 빌드와 배포를 자동화한 절차
CNAMT	Cloud Native Application Migration Toolkit. 이 백서가 애플리케이션 현대화 분석 도구로 다루는 정적 분석 도

용어	정의
	구
CNTA	Cloud Native Transformation Assessment. 클라우드 네이티브 전환 사전 진단
EJB	Enterprise JavaBeans. 서버 측 업무 컴포넌트 규격
IaaS	Infrastructure as a Service. 서버와 스토리지 같은 인프라 자원을 빌려 쓰는 서비스
ISP	Information Strategy Planning. 정보화전략계획
JDK	Java Development Kit. Java 개발·실행 도구 모음
JMS	Java Message Service. 메시지 연계 규격
JNDI	Java Naming and Directory Interface. 이름으로 자원을 찾는 표준 인터페이스
JNI	Java Native Interface. Java에서 C 등으로 작성한 네이티브 코드를 호출하는 인터페이스
JSP	JavaServer Pages. 화면을 만드는 서버 측 페이지
JVM	Java Virtual Machine. Java 가상 머신
KubeVirt	Kubernetes에서 VM을 실행하는 오픈소스 기술. 예외 VM을 컨테이너로 옮기기 전까지 임시로 받는 브릿지 기술
Kubernetes	컨테이너의 배치와 운영을 자동화하는 오픈소스 플랫폼
Mandatory	고치지 않으면 빌드나 실행이 되지 않는 이슈 분류
M/D	Man/Day. 한 사람의 하루 작업량
M/M	Man-Month. 한 사람이 한 달 동안 수행하는 작업량. 본문은 월 20일 기준으로 환산
MSA	Microservices Architecture. 애플리케이션을 독립 배포가 가능한 작은 서비스로 나누는 구조
NIA	한국지능정보사회진흥원
OOMKilled	메모리 한도 초과로 컨테이너가 강제 종료됐을 때 Kubernetes가 기록하는 상태
Optional	고치지 않아도 동작하지만 최적 상태가 아닌 이슈 분류
PaaS	Platform as a Service. 실행 환경까지 제공받아 쓰는 서비스

용어	정의
PoC	Proof of Concept. 개념 검증
Potential	필수 여부를 판단할 정보가 부족해 사람의 검토가 필요한 이슈 분류
Refactor(재구성)	변경 정도에 따라 일부 개정이나 전면 재개발이 필요한 전환유형
Rehost(단순전환)	시스템 소프트웨어 버전 변경 정도로 옮기는 전환유형
Replatform(플랫폼전환)	OS, WEB·WAS, DBMS 같은 플랫폼 변경으로 소스 수정과 데이터 변환이 따르는 전환유형. 컨테이너 전환이 속하는 유형
Story Point	분석 도구가 이슈마다 수정에 드는 노력을 0·1·3·5·7·13점으로 매긴 점수
TCO	Total Cost of Ownership. 총소유비용
U2L	Unix to Linux. Unix에서 리눅스로 옮기는 전환
VM	Virtual Machine. 게스트 OS를 포함해 하이퍼바이저 위에서 실행되는 가상 머신
WAS	Web Application Server. 웹 애플리케이션 서버

클라우드 네이티브 전환 사전 진단(CNTA)

CONTACT

WEB

msap.ai

www.msap.ai/

EMAIL

hello@msap.ai

TEL

02-6953-5427

0269535427

YOUTUBE

@msaptv

www.youtube.com/@msaptv

LINKEDIN

linkedin.com/showcas...

www.linkedin.com/showcase/msap-ai/

FACEBOOK

facebook.com/opennaru

www.facebook.com/opennaru



SCAN