제 8 장 최상의 통합 방법

이 장의 목적은 WebSphere Product Center 구현에 있어 최상의 통합 방법을 요약하는 것입니다. 이러한 최상의 교범을 사용하면 시스템간의 고품질의 종속 가능한 통합을 성취하는 데 도움을 줍니다. 통합의 모든 측면을 다루기 위해 이 문서는 통합의 이러한 각기 다른 측면의 각각과 연관된 최상의 교범을 구별하도록 개발되었습니다.

통합의 핵심 요소는 다음과 같습니다.

- 설계 원칙
- 구현
- 유효성 검증
- 표시 여부

정의 및 약어

통합 특성: 아래에 나열된 특성을 사용하여 WebSphere Product Center 구현에서 발생하는 다양한 구현 유형을 이해할 수 있습니다. 나머지 문서에서는 구현의 크기 또는 종료에 적용 가능한 최상의 교범 또는 지침을 강조 표시합니다.

소스 또는 대상 시스템으로서의 WebSphere Product Center

가장 명백한 특성은 WebSphere Product Center가 교환 중인 정보에 대한 소스 시스템인지 또는 대상 시스템인지 여부입니다. 소스 시스템은 통합에 해당 제한조건을 두며, 그 중 가장 중요한 것은 (a) 델타 배급을 수행하는 기능, (b) 통합을 시작하는 기능, (c) 전송된 데이터의 완료/실패에 대한 통지를 수신하고 적절한 조치를 취하는 기능 및 (d) 지원되는 프로토콜과 형식과 EAI 하부 구조의 지원입니다.

제어 시스템

통합의 내부 트리거에 따라 조치를 취하는 시스템으로 제어 시스템을 정의하게 됩니다. 예로는 스케줄에 따라 작업으로 배급을 실행 중인 WebSphere Product Center가 있습니다. 또 다른 예로는 항목 추가 결과로서 WBI 어댑터를 트리거하는 SAP가 있습니다. WebSphere Product Center가 통합의 소스 또는 대상 시스템이든지 통합에서 제어 시스템인 시스템과는 완전하게 독립적입니다. 몇 가지 경우가 가능합니다. FTP 서버 또는 EAI 도구와 같은 중간 단계가 포함되면, 소스 및 대상 시스템은 모두 제어 시스템이 될 수 있습니다. 스케줄에 따라 레거시 시스템은 FTP 서버에 파일을 저장하는 반면, 스케줄에 따라 WebSphere Product Center는 파일을 가져옵니다. WebSphere Product Center가 제어된 목적지 시스템(데이터 가져오기를 트리거하기 위해 외부의 어떤 것을 대기)인 위치의 예로는 Transora의 항목 갱신이 말하려는 메시지 내용과 함께 호출자를 통해 WebSphere Product Center에 메시지를 게시하는 IBM WBI가 있습니다. WebSphere Product Center가 제어된 소스 시스템(데이터 내보내기를 트리

거하기 위해 외부 시스템 대기)인 위치의 예로는 가져올 준비가 되어 있는 파일이 있는지 보기 위해 정기적으로 IBM WBI가 WebSphere Product Center 대기열을 폴하는 위치가 있습니다.

프로토콜

프로토콜, 형식 및 메시지 대 파일 기반 통합 간에 WebSphere Product Center 구현 팀과 고객 자원에 많은 혼란이 있습니다. 따라서 이 문서의 목표는 이러한 개념에 대해 공통적인 명칭을 설정하는 것입니다. 프로토콜의 예로는 FTP(File Transfer Protocol), HTTP(Hyper Text Transfer Protocol), SMTP(Simple Message Transfer Protocol, 전자 우편), JMS(Java Messaging Service) 및 IBM WebSphere Message Queuing(IBM WebSphere MQ)이 있습니다. 프로토콜은 엔벨로프와 같은 것, 번호 및 예상된 응답과 같은 인코딩을 정의하나 전송되고 있는 내용과는 아무런 관계가 없습니다. 모든 통합에서는 항상 최소한 하나가 있으므로 사용되고 있는 프로토콜에서 완전히 지원해야 합니다. 또한, 통합의 여러 단계는 실제로 여러 프로토콜을 사용 중일 수도 있습니다. SAP의 WBI 어댑터는 HTTP를 통해 SAP에서 WBI ICS(Inter Connection Server)로 데이터를 전송 중일 수 있으며, WebSphere Product Center가 MQ 클라이언트로서 연결된 또 다른 대기열 관리자에게 전송할 해당 데이터를 IBM MQ 대기열 관리자에게 넘겨줍니다.

형식

데이터가 배치된 형식은 프로토콜과는 관계가 없습니다. 형식의 예로는 CSV(Comma Separated Value), 파이프 구분, XML(eXtensible Markup Language) 또는 EDA(Electronic Data Interchage) 메시지와 같이 일부 사전 정의된 필드 및 레코드 구조가 있습니다. 각 형식은 위치/길이 매개변수 또는 태그를 통해 필드를 정의합니다. 특정 형식에 필요할 수도 있는 인코딩을 기억하는 것이 중요합니다. 예를 들면, 꺾쇠 괄호(<', '>')와 같은 문자는 XML에 없어야 한다거나 내용에 실제로 CSV의 쉼표가 포함될 수 있다는 사실을 구현할 때 자주 잊어버리게 됩니다.

데이터 크기

이 크기는 매우 자주 "메시지 기반" 또는 "파일 기반" 통신과 혼동되므로 이 권한을 가져오는 것이 중요합니다. "메시지 기반" 통합은 일반적으로 더 작은 데이터 교환 및 다음과 같은 등록 정보와 관련되어 있는 것으로 가정됩니다.

- 데이터는 변경사항이 내보내기/가져오기가 매주 발생할 수 있는 종래의 "일괄처리" 지향 시스템보다 훨씬 더 작은 어커런스 시간 간격 내에서 통신되도록 더 빈번하게 및 더 작은 양으로 교환되고 있습니다.
- 두 시스템(소스 및 대상)은 서로 연결되어 있어 처리를 위해 가져오기 전에 FTP되거나 한 주 동안 파일 시스템에 저장되어 있을 수 있는 생성되고 있는 파일이 아니라 전송된 메시지가 처리되고 있으며 다시 수신확인됩니다.

그러나 메시지 기반을 파일 기반 또는 일괄처리 통합과 구별하기 위해 그릴 수 있는 명확한 선이 없으므로, 명백한 특성을 정의하고 혼동하거나 겹

치지 않도록 하는 것이 중요합니다. 따라서 전체 데이터 크기는 "메시지 기반" 또는 "일괄처리" 통합으로 레이블이 지정되어 있는지 여부보다 고려해야 하는 더 중요한 특성이어야 합니다.

통신 유형

고려할 또 다른 크기는 통합과 관련될 통신 유형입니다. 동기 통신은 특정 조치의 결과로 사용자나 시스템에 직접 피드백을 제공합니다. 예를 들어, 통신에 HTTP를 사용하면, 조치가 게시된 후 시스템 또는 사용자에게 자동 피드백이 제공됩니다. 이와는 반대로 비동기 통신은 "fire-and-forget" 전략을 더 사용합니다. 통합이 예를 들어 시스템에서 가져올 FTP 서버의 파일 저장과 관련되면, 해당 조치의 결과로 파일을 저장하는 시스템에 대한 자동 피드백이 없습니다.

빈도

"데이터 크기"와 함께 이 "빈도" 크기는 주기적으로 처리되어야 하는 총 데이터 양을 캡처합니다.

통합 스레드

이 중간 시스템 및 하부 구조 크기는 EAI 하부 구조가 사용되고 있는지 여부에 관계없이 캡처합니다. 또한 레거시 시스템과 통합 시, 중간 프로그램은 레거시 시스템으로 데이터를 업로드 또는 추출하도록 작성될 수 있습니다. 이러한 중간 시스템 또는 프로그램은 통합 체인에서 대부분 가장 약한 링크이므로 이러한 중간 시스템 또는 프로그램을 이해하고 문서화하는 것은 중요합니다.

특히 다중 홉을 필요로 하는 복잡한 통합에서(예: 목적지 시스템에 대한 WBI에 WebSphere Product Center) 비표준은 (직접 데이터베이스 갱신과 같은) 다중 프로토콜 또는 기타 통신 인증 확인(방화벽을 통한 통신과 같은)을 의미하며 이른 통합의 작업 중인 단일 스레드 또는 완전한 경로를 설정합니다. 이러한 연결성 문제를 동시에 해결하려면 문제를 판별하여 다른 부분(예: 네트워크 관리 또는 IBM WBI에서 작동하는 팀)에 충분한 시간을 제공합니다.

위에 나열된 통합 특성은 WebSphere Product Center 구현에서 통합을 설명하는 표준 용어가 됩니다. 분석/설계 단계에서 PS 팀에서 제공한 문서는 이러한 크기를 명백하고 일관성있게 사용해야 합니다.

약어

약어	정의
EAI	엔터프라이즈 응용프로그램 통합
FTP	파일 전송 프로토콜
HTTP	하이퍼 텍스트 전송 프로토콜

MQ	IBM의 메시지 대기열 미들웨어. 모든 연결 솔루션이 현재 WebSphere 브랜드에 속하기 때문에 흔히 IBM WebSphere MQ를 가리킵니다.
ICS	IBM의 WBI Inter Connect Server
WBI	IBM의 WebSphere Business Integration suite, IBM의 EAI suite.

설계 원칙

재사용 가능성

통합 구현 방법의 기초를 이루는 전체적인 토대는 재사용 가능성입니다. WebSphere Product Center가 커지고 더 많은 클라이언트 구현이 수행되면, 이전에 통합되지 않은 시스템과 이전 구현으로 통합된 시스템 둘 모두와의 통합을 신속하게 확장하고 해결할 수 있어야 합니다. 이러한 요구사항을 해결하려면, 최고의 효율성으로 그렇게 수행할 수 있는 또 다른 클라이언트에 대해 동일한 시스템과 통합해야 할 경우 재사용 가능성으로 모든 통합 결과에 접근해야 합니다.

재사용 가능성은 (a) IBM WBL 및 일반 비즈니스 오브젝트의 해당 모델과 같은 EAI 도구 강화, (b) 데이터 모델과 독립적인 형식 선택, (c) 데이터 모델과 독립적이며 다른 구현에서 재사용할 수 있는 WebSphere Product Center 스크립트의 라이브러리 작성(수신확인, 폴링 등)을 통해 실행됩니다.

정보 공유

통합 수단으로 통신

개념적으로 통합은 단순히 제어 시스템 WebSphere Product Center과 제어된 시스템 간의 통신으로 트리거될 수 있는 일련의 이벤트로 간주될 수 있습니다. 이러한 이벤트는 시스템, 내용 또는 파일에 대해 폴하는 자동화된 프로세스 또는 다른 기본 통신 방법 사이에서 전달되는 메시지를 통해 트리거될 수 있습니다. 통신에는 변경된 유형(추가, 갱신, 삭제), 고유 통신 ID(추적/확인용) 및 WebSphere Product Center 또는 완전한 시스템 내에서 변경해야 하는 관련 내용이 포함됩니다.

신뢰도 측정

변경사항을 통신하기 위해 시스템 간에 정보를 전달하는 것외에도, 특정 트랜잭션의 완료 또는 실패를 통신할 적절한 방법도 있어야 합니다. 그러한 데이터 교환 통신은 통신의 동기 양식으로 가장 즉각적으로 구현할 수 있으며, 통합 시스템이 다른 한쪽 끝의 실패한 수신으로 인해 특정 트랜잭션을 다시 전송해야 할 수 있는지 추적할 수 있게 하므로 통합의 신뢰도가 향상되며 궁극적으로는 보증됩니다.

정보 형식

이러한 통신의 특정 형식은 구현 시 재사용할 수 있는 주변의 형식 및 처리 기능 둘 모두에 충분한 일반적인 형식으로 설계되어야 합니다.

시스템 사이의 통신에 사용할 수 있는 일반 형식을 고려해 볼 때, 다음 요구에 만족하는 형식의 유용성을 기억하는 것이 중요합니다.

- 국제 문자 세트 및 특수 문자(쉼표, 따옴표, 꺾쇠 괄호 등)
- 복잡한 구조(예: 내용 및 관계 계층 구조)
- 내용 또는 항목 위치 표시기의 여러 인스턴스를 인스턴스당 여러 값으로 처리할 수 있음

정보 처리

시스템 사이에서 전송되고 있는 정보 형식을 어느 정도 일반적인 것으로 간주할 수 있는 경우 모든 구현이 사전 정의된 형식에 적합한 것은 아닙니다(특히, WebSphere Product Center와 완전한 시스템 사이의 직접 링크로서 통합을 볼 수 있는 경우). 데이터 모델과 같은 차이로 인해 모든 구현에서 형식과 WebSphere Product Center 스펙 사이의 형식 및 맵핑의 재툴링 요구를 피하려면, XML 형식과 WebSphere Product Center 스펙 사이의 재사용 가능한 맵핑 기능을 사용할 것을 권장합니다.

EAI 플랫폼 사용

이를 수행하는 한 가지 방식은 WBI 또는 webMethods 모음과 같은 EAI 플랫폼을 사용하는 것이며, WebSphere Product Center가 하나의 완전히 재사용 가능한 메시지 형식(예: 단일 XML DTD)을 통해 통신할 수 있는 재사용 가능한 커넥터를 빌드할 수 있게 합니다. 구현의 특정 사항으로 인해 발생하는 차이점은 정보를 처리하기 위해 WebSphere Product Center 기능을 재툴링하지 않고 WBI에서 변환될 수 있습니다. 필요한 WebSphere Product Center 기능을 재툴링하지 않고 구현을 통해 동일한 기능을 사용할 수 있습니다.

기타 옵션

그러나 고려할 또 다른 요소는 그 엔터프라이즈를 통해 다른 시스템에서 이미 사용 중인 형식을 특정 클라이언트에서 다시 사용해야 할 수도 있다는 점입니다. 이런 경우 WebSphere Product Center에서는 이미 존재하는 DTD를 사용하기 위해 WebSphere Product Center가 아닌 엔터프라이즈의 다른 시스템에서도 이해하도록 변환되어야 할 완전히 별개의 DTD를 도입하기 어려워집니다. 이러한 경우, WebSphere Product Center와 DTD 내의 스펙 사이를 변환하기 위해 재사용 가능한 기능을 사용해야 합니다.

사용 중인 EAI 플랫폼에서도 이와 같이 될 수 있으며, 이 접근 방법은 정보 통신을 위해 내부 DTD에 대한 WebSphere Product Center 관리 정보 맵핑을 이해하는 측면에서 특정 클라이언트에 더 유용할 수 있습니다.

이벤트 처리

WebSphere Product Center 내의 자동화된 프로세스는 이상적으로 이벤트를 처리합니다. 예를 들어, WebSphere Product Center 릴리스에 도입된 대기열 기능은 메시지 전송(아웃바운드 대기열)과 응답 및 수신 메시지(인바운드 대기열) 수신을 모두 처리하는 데 사용될 수 있습니다. 그러면 대기열 처리 스크립트는 메시지의 실제 처리를 처리하는 데 사용될 수 있으므로 실제로 특정 메시지의 결과로 트리거될 이벤트를 수행할 수 있습니다.

이벤트 처리는 특정 기능이나 특정 WebSphere Product Center 버전에 직접 연결될 필요는 없습니다. 그러나 기타 이벤트 처리 방법에는 FTP 서버를 폴하는 WebSphere Product Center 내의 스케줄된 작업, 파일의 로컬 파일 시스템을 확인하는 스케줄된 작업(문서 저장소를 통해), 게시된 정보에 따라 호출자 기반 트리거 스크립트 작동 이벤트 소유 또는 기타 방법이 포함될 수 있습니다. 궁극적으로 특정 통합의 데이터 크기 및 빈도 크기 요구사항을 주의 깊게 고려하여 방법을 선택해야 합니다.

변경사항 추적

시스템 사이의 완전한 동기화를 구현하려면, 완전한 시스템에 전달할 때 더욱이 효과적으로 태그가 지정될 수 있거나 그렇지 않을 수 있는 내용 및 항목의 변경사항을 추적하는 방법이 WebSphere Product Center에 있어야 합니다. 예를 들면, WebSphere Product Center(소스 시스템으로) 내에서 항목이 삭제되면, 동일한 항목을 삭제하도록 대상 시스템에 통지하는 대상 시스템으로 전송되고 있는 메시지를 트리거할 수 있을 뿐만 아니라 WebSphere Product Center가 항목이 실제로 대상 시스템에서 삭제되었는지 여부를 인식할 수 있도록 특정 통신의 완료 또는 실패를 추적할 수도 있습니다.

재사용 가능한 커넥터

커넥터 저장소

구현 과정 및 협력이 발전함에 따라 점차적으로 다양한 시스템으로 재사용 가능한 커넥터의 저장소를 빌드하게 됩니다. 가능할 때마다, 이러한 커넥터를 통해 이동하는 처리 항목 등의 기능을 특정 구현 관점에서 거의 수정하지 않거나 수정하지 않고 재사용할 수 있으므로 이러한 커넥터 재사용에 온갖 노력을 기울여야 합니다. 물론 총괄적으로 구현 실행 시간의 속도가 상당히 빨라지며, 문제를 발견하여 시간에 따라 해결할 때 커넥터의 전체 신뢰도 및 안정성과 이러한 커넥터를 사용하는 구현이 향상됩니다.

아직 커넥터가 정의되지 않은 시스템과 통합할 때, 통합 전문가는 특정 구현에 대한 통합에 모두 사용될 수 있는 재사용 가능 커넥터를 신속하게 빌드할 수 있어야 하며, 다른 구현에서 시스템과 통합해야 할 경우 나중에 사용하기 위해 커넥터 저장소에 저장할 수도 있습니다.

커넥터 사용법

커넥터는 수행되어야 할 수 있는 모든 수정이 시스템 사이에서 전달되고 있는 정보의 EAI 레이어 처리 변환을 통해 수행되도록 사용해야 합니다. 다시 말해, EAI를 통해 전달된 정보를 처리하기 위해 WebSphere Product Center 내에 재사용 가능한 기능을 다시 작성하기 전에, WebSphere Product Center 내의 기능을 재작성할 필요가 없도록 모든 필수 변환을 수행할 수 있는 EAI 플랫폼의 기능을 사용해야 합니다.

구현

구현 스케일링

소규모 통합

전체 통합의 대형 타스크는 더 작고, 더 쉽게 관리할 수 있는 타스크로 구분되어야 합니다. 예를 들면, 훨씬 더 작은 통합으로 하나의 완전한 통합을 구분하여 수행될 수 있습니다 – 각 항목 유형(스펙)마다 "별도의" 통합에서부터 각 컨테이너(카탈로그)마다 통합까지 속성 그룹의 통합 밑에 있는 모든 방법(필요한 경우). 이러한 각각의 "소규모 통합"이 완전하게 작동한다는 확신이 있으면, 하나의 완전한 통합을 구성하기 위해 조합할 수 있습니다.

기능 세분화

시스템 사이의 통합이 발생해야 하는 레벨에는 주의를 기울여야 합니다. 예를 들어, 대상 시스템으로 변경사항을 전송할 때 특정 날짜 이후의 모든 변경사항, 마지막 변경사항이 전송된 이후에는 특정 카탈로그의 변경사항만, 특정 항목 그룹에 발생한 변경사항만 또는 모든 항목을 통해 특정 속성에 발생한 모든 변경사항을 전송할 수 있습니다. 특정 요구사항은 구현에 따라 달라지나 요구가 적절하게 제공될 수 있도록 구현의 설계 프로세스에서 필요한 세분화를 고려하는 것이 중요합니다.

성능 조정

일반 성능 주석

결과적으로 성능 문제를 방치하지 마십시오. 게임에서 형식 및 최신의 다른 통합 관점을 변경 및 수정하는 것이 더 쉬우나 성능 병목 현상은 기본적으로 재설계되어야 하며 때로는 기술적 지원이 필요할 수 있습니다. 적절한 개발 과정에서 스크립트에 성능 측정 후크를 저장하십시오.

성능 측정

소규모 통합 접근 방법을 가정하면(구현 섹션에서 설명한 대로), 성능은 소규모 통합 타스크마다 필요한 총 시간을 측정하

여 통합의 각 단계에서 측정해야 합니다. 그러면 가능한 낮은 성능 영역이 적절하게 세분화된 레벨에서 식별될 수 있으므로 훨씬 더 쉽게 성능 변경에 목표가 될 수 있습니다.

성능 변경

일단 성능 문제점 영역이 식별되면, 느린 조작의 근본적인 원인을 판별하기 위해 세부 분석이 수행되어야 합니다. 세부 분석은 WebSphere Product Center의 미들웨어 프로파일링 및 작업 세부 화면의 성능 탭과 같은 도구를 사용하여 수행될 수 있습니다. 그런 후 분석은 스크립트 또는 SQL 조회의 특정 영역에 초점을 맞추기 위해 적용될 수 있으며 스크립트 수정 또는 재작성이나 데이터베이스 조회를 향상시키기 위한 기술 포함 등의 적절한 조치를 취할 수 있습니다.

유효성 검증

안정성

소규모 통합의 장점

소규모 통합을 구현하면(구현 섹션에서 설명한 대로) 통합이 작동되는 것으로 표시된 모든 영역의 세부 목록을 제공하여 통합이 완료된 훨씬 더 높은 레벨의 신뢰도가 제공되어야 합니다. 소규모 통합의 가시성 없이 작동 중인 통합의 세부사항을 증명하는 것이 더 어려울 뿐만 아니라 전체적으로 통합에는 식별, 진단 및 디버그하기 어려운 문제가 발생할 수 있습니다. 소규모 통합을 구현하면 전체적인 통합 안정성이 향상됩니다.

확장 가능한 테스트

소규모 통합의 장점

소규모 통합을 구현하면(구현 섹션에서 설명한 대로) 발생한 모든 오류 또는 문제점이 대량의(잠재적으로 무의미한) 복잡도로 인해 희미해지지 않도록 훨씬 더 세부화된 레벨에서 통합 테스트가 발생할 수 있습니다. 따라서 위에서 언급한 대로, 발견된 문제 진단, 디버깅 및 해결 프로세스는 이 접근 방법으로 인해 모두 현저하게 속도가 빨라집니다.

대표 환경 대 완전한 환경

통합 테스트는 최종 환경(동일한 스펙, 유효성 검사 규칙, 값 규칙, 보기)과 동일한 구성으로 그러나 가능한한 대표 엔티티(로케일, 카탈로그, 카테고리 트리, 항목, 카테고리, 조직, 사용자 및 역할)는 적게 대표 환경에서 수행되어야 합니다. 이렇게 하면 테스트 실행, 화면 로드에 소요되는 시간이 줄어들며 일반적으로 완전히 채워진 환경의 테스트에 비해 테스트에 소요되는 시간의 속도가 빨라집니다. 모든 테스트 및 디버깅은 이

환경에서 수행되어야 합니다.

대표 환경 및 모든 것에서 테스트가 완료된 경우에만 완료되어 완전히 채워진 환경에서 통합이 검증된 경우 거기에 작동 중인 것으로 나타납니다. 그러나 대표 환경에서 우연히 무시한 위험한 시나리오가 없는지 확인하고 통합의 프로덕션 레벨 성능을 테스트하기 위해 이 단계는 여전히 실행되어야 합니다.

확장 가능한 프로세스 테스트

스케줄 가능한 작업(예: 가져오기, 내보내기)은 먼저 매우 작은 수의 대표 항목(10 이하)에서만 실행되어야 합니다. 이 숫자는 실제로 이러한 항목을 처리 중인 스크립트에서 수집된 신뢰도 레벨에 비례하여 증가해야 합니다. 이 접근 방법은 전체 프로세스가 실행 중인 처음 몇 분 내에 곧 실패하지 않고 결국 어떤 것이 잘못 이동했다는 것을 찾기 위해 프로세스를 실행 중인 개인만이 시간에 관한 문제로 인해 대량 작업을 실행하지 않도록 합니다.

작업과 연관된 스크립트의 조작을 완전히 신뢰한 후에만 완전한 데이터 세트와 관련된 작업을 실행해야 합니다. 완전한 환경 권장사항에서와 마찬가지로, 우연히 무시한 위험한 시나리오가 없는지 확인하고 작업의 프로덕션 레벨 성능을 테스트하기 위해 이 단계는 여전히 실행되어야 합니다.

표시 여부

보고

소규모 통합의 장점

소규모 통합을 구현하면(구현 섹션에서 설명한 대로) 더 작고 더 빠른 구현 가능한 통합으로 인해 더 자세한 보고 레벨을 사용할 수 있습니다. 전체 통합 레벨에서 구현 진행 시 보고와 비교해 볼 때 이러한 보다 세부적인 보고 레벨은 더 명확한 구현의 수량 추적에 사용할 수 있습니다.

소규모 통합은 나열될 수 있고 완전한 통합에서 더 큰 그림과의 관계는 차트에 자세하게 나와 있으며 전체 구현 진행의 정확한 그림이 소규모 통합 타스크 진행의 보고로부터 쉽게 그려질 수 있습니다.

소유권

여러 팀으로 작업할 경우에도, 통합에서는 하나의 개인에게 소유권을 지정하십시오. 이 개인의 작업은 단일 스레드가 먼저 설정되어 있는지, 팀이 이 문서 지침에 따라 작동하고 있는지, (a) 소규모 통합, (b) 확장 가능한 프로세스 테스트 및 (c) 대표 환

경 대 완전한 환경을 통한 증가하는 빌드/테스트 주기가 다양한 팀에서 동기화되는지 확인하는 것입니다.

문서

형식 및 접근 방법을 명확하게 식별

통합에서 작동 중인 여러 팀이 있을 때 실행할 명확한 경로를 결정하고 사용할 모든 형식을 명백하게 문서화하십시오. 대부분의 공통적인 예로는 WebSphere Product Center 팀이 WebSphere Product Center에서 데이터를 내보낼 때 작동 중인 위치와 고객 또는 SI 팀이 목적지 시스템으로 데이터를 업로드할 때 작동 중인 위치가 있습니다. 공통 형식의 스펙없이 작업을 시작하지 말고 매일 최신으로 이 문서를 보관하십시오. 프로젝트 관리자가 수행해야 하는 절대적인 요구사항입니다.

이 접근 방법은 대표 환경 사용 및 소규모 통합 수행과 일치합니다. 안정되고 가시적인 진행을 확실히 하려면, 두 팀 모두 점진적으로 빌드 및 테스트해야 합니다.

WebSphere Product Center 통합을 위한 최상위 10개 지침

명확한 공통 용어를 사용하여 통합 설명

모든 구현은 통합 특성 섹션에 식별된 크기를 사용해야 합니다.

재사용 가능성

확장의 핵심은 이전 통합의 학습과 재사용 가능성 섹션에 설명된 재사용 가능성 원칙을 기억하는 통합 패키징에 있습니다.

표시 여부

보고 진행의 전체 메트릭을 설정하고 프로젝트 관리자에게 아무리 나빠도 수 일마다 명확한 상태 갱신을 제공하십시오.

소규모 통합

해당 통합을 인식하는 다양한 크기(카탈로그, 속성)에 따라 대형 통합의 복잡도를 분할하십시오. 한 번에 하나의 소규모 통합에 초점을 두고 가시성 메트릭에 직접 연결하십시오.

대표 환경 대 완전한 환경

디버그 및 테스트하기 쉬운 대표 환경을 유지보수하십시오. 스크립트 및 스펙의 유효성 검증을 신뢰할 수 있는 경우에만 완전한 환경으로 이동하십시오. 가시성 메트릭에 연결하십시오.

확장 가능한 프로세스 테스트

모든 작업을 작은 데이터 세트로 테스트하여 완전한 데이터 세트로 확장하기 전에 정확도를 확인하십시오. 가시성 메트릭에 연결하십시오.

성능

논리 또는 형식화의 정확성에 대해 걱정하지 말고, 개발 환경에서 그리고 이후에 문제를 식별하기 위해 규칙적으로 초기에 일부 성능 테스트를 실행하십시오.

초기에 단일 스레드 설정

특히 여러 홉, 여러 프로토콜 또는 비표준 방법을 필요로 할 수 있는 복잡한 통합에서 작동 중인 단일 통합 스레드를 초기에 설정하십시오.

스펙 및 문서 설계

특히 통합에서 작동 중인 여러 팀이 있을 때 실행할 명확한 경로를 정의 및 문서화하고 사용할 모든 형식을 명확하게 문서화하십시오.

단일 소유자

여러 팀으로 작업할 경우에도, 통합에서는 하나의 개인에게 소유권을 지정하십시오.

EAI 플랫폼 통합

접근 방법

일반 통신 형식

가능할 때마다 일반 통신 형식은 이전 프로젝트에서 설계되거나 재사용되어야 합니다. 형식이 일반적일수록 모든 시스템이 서로 통신하는 데 필요한 형식을 특별히 재작동하지 않고 통합에 더 많은 시스템이 포함될 수 있습니다. 물론 형식이 더 일반적이 되는 성능에서는 번갈아 사용할 수 있으므로, 한 프로젝트에 올바른 형식이 다른 프로젝트에서는 이상적인 선택이 아닐 수도 있습니다. 사용할 특정 형식을 판별할 경우 여전히 통합 특성을 고려해야 합니다.

컨텐츠 맵핑

가능한한 많이 WebSphere Product Center 내의 내용 모델과 통신 형식을 통해 표시된 모델 사이의 맵핑은 동적으로 갱신 가능한 방법을 사용하여 수행되어야 합니다. 통합 특성의 조사에 따라 특정 프로젝트 요구에서는 이러한 맵핑 작성이 완전히 동적으로 갱신될 수 없다고(예: 절대적인 최대 처리량의 높

은 우선순위로 인해) 규정할 수 있습니다. 이를 수행하는 한 방법은 WebSphere Product Center 내 모델에서 카테고리 트리의 특정 노드가 맵핑된 속성의 스펙 노드 경로를 나타낼 수 있는 카테고리 트리(예: XML 구조 표시)를 이 트리와 관련된 단일 노드 스펙에 사용하는 것입니다. 그러면 반복적인 스크립트는 이 카테고리 트리와 정의된 맵핑에 따라 XML 파일로의 항목 맵핑을 처리하는 데 사용될 수 있으며 적은 노력으로 중첩된 다중 어커런스를 제공할 수도 있습니다.

추가 장점

정보 번역/변환

통합과 관련된 시스템에서는 통합에서 기타 시스템의 정보 또는 내용 제한사항 및 요구사항 자체를 처리할 필요가 없습니다. 내용 번역 및 변환을 처리하는 데 EAI 플랫폼을 손쉽게 사용할 수 있습니다. 예를 들어, WebSphere Product Center가 FLAG의 값을 "TRUE" 또는 "FALSE"로 저장하면 통합된 시스템은 값을 "Y" 또는 "N"로 저장할 수 있습니다. EAI 플랫폼은 WebSphere Product Center가 항상 TRUE/FALSE를 전송하고 TRUE/FALSE가 전송된다고 가정할 수 있는 반면, 통합 시스템에서는 항상 Y/N가 전송되며 전송될 것이라고 가정할 수 있습니다. 이로 인해 회선은 중단되며 통합에 추가 시스템이 포함되면 이 추가 시스템 입장에서는 레코딩이 필요없게 됩니다.

클라이언트 이해

클라이언트에서 이미 익숙해져 있는 플랫폼을 다시 사용할 수 있으므로, 클라이언트는 통합에서 알려진 기능을 사용하는 추가 신뢰도를 얻게 됩니다(예: EAI 플랫폼). 추가로 클라이언트 특정 통신 형식이 이미 존재하며 WebSphere Product Center 통합에 다시 사용되면, 클라이언트측 개발자에게는 WebSphere Product Center가 맵핑될 통신 형식을 이해하기 위한 추가 훈련이 필요없습니다.

통신 유연성 및 신뢰성

대부분의 EAI 플랫폼에는 다양한 프로토콜에서 통신이 발생할 수 있으며 통신이 브로커에 의해 전달되도록 하는 기본 기능이 있습니다. 이로 인해 WebSphere Product Center는 통신하는 데 필요한 문서 생성에 초점을 두고 이 문서를 다양한 시스템으로 전달하기 위한 잠재적으로 서로 다른 방법 지원에 대해 걱정할 필요가 없으며 각 시스템에서 문서를 수신했는지 여부 추적에 대해 걱정할 필요도 없습니다. 여기에는 EAI 레이어 및 플랫폼이 관련되며 WebSphere Product Center에서는 전체적인 통합 스레드 관점에서 이들을 인식하기만 하면 됩니다.