

Ikasan Enterprise Integration Platform Architecture

Robust and Adaptable Enterprise Integration

Ikasan Enterprise Integration Platform Architecture: Robust and Adaptable Enterprise Integration

0.8

Table of Contents

1. Introduction	1
Overview	1
2. Architectural Strategies	2
Commonality	2
Flexibility	2
Robust and Guaranteed Operation	2
Single Point of Integration	2
Separation of Concerns	2
Application Connectivity	2
Application Data Constructs	3
Common Data Operations	3
Business Event Services	3
Loose Coupling	3
Tight Cohesion	3
Standard Open Technologies	3
Standardisation through a Platform	3
Modularity	3
Extensibility	4
Re-useability	4
Inversion of Control	4
Commoditised Solution	4
3. Concepts	5
Integration Concepts	5
System Integration	5
Data Integration	6
Event Integration	6
Event/Service Presentation	7
Integration Layer View	7
TODO - Ikasan Design	7
Ikasan Design Overview	7
Ikasan System Integration	8
Ikasan Data Integration	8
Ikasan Event Integration	8
Ikasan Event Presentation	8
Index	9

Chapter 1. Introduction

This document is one of a suite of documents describing the Ikasan Enterprise Integration Platform, an open source integration platform for robust and adaptable integration solutions.

Overview

The Ikasan Enterprise Integration Platform (EIP) addresses the problem domain most commonly known as Enterprise Application Integration (EAI). Enterprise application integration can be, and already has been, approached a number of different ways by a number of projects/vendors, both Open Source and closed commercial frameworks. It is the intention of the Ikasan Enterprise Integration Platform to address this domain as commoditised configurable solutions rather than another development framework.

This document describes the central premise defining Ikasan; the guiding strategies which have influenced Ikasan's design; and the architectural concepts which shape the integration solution approaches.

Ikasan was originally born out of addressing integration issues within Financial Services where guaranteed once-only data delivery is paramount. The central premise for Ikasan can simply be defined as “the provision of robust and adaptable integration solutions which expose the business artefacts, whilst fencing-in the application's specifics”. The intention here is for Ikasan to take care of the peripheral issues of integration, such as communication and data transformation, and allow the presentation of business entities through common constructs and services thus allowing the user to focus on the core issue of business data orchestration.

Where possible every effort has been made to ensure this document remains free from technical detail thus keeping it open to as wide as an audience as possible.

Chapter 2. Architectural Strategies

This chapter details the strategies laid out as guidelines for the design of the Iklan Enterprise Integration Platform.

Commonality

The nature of an integration platform is such that the integration requirements for one application are likely to be repeated, either in part or whole for subsequent application integration solutions. It would seem foolish to implement the same artefacts over and over again; or, in the case where requirements differ only slightly, distort the implementation of one to meet the requirements of others. This approach would not require many application integration solutions before the implementation became an unstable, tightly coupled mash of technologies. The requirement to provide similar services across a range of solutions must result in a common approach for each solution. This common approach must allow each integration solution's requirements to be met, but in a standard approach utilising and extending existing common core functionality.

Flexibility

A common approach must allow for the integration of unknown entities and the provision of new services to support the future solutions. The integration of applications must be undertaken within a controlled, structured manner utilising common functionality, however, this cannot result in a rigid inflexible platform. The ability to adapt and accommodate new requirements is fundamental to the success of any integration platform.

Robust and Guaranteed Operation

All artefacts within the platform must operate within the bounds of maintaining delivery integrity. This ensures artefacts participating in any operation remain in a consistent state relative to each other regardless of the success or failure of that operation. This aspect of integrity can subsequently be leveraged to support automatic recovery and retry for artefacts within the platform. Without these operational bounds the platform will not be robust and cannot guarantee business data transport or coordination.

Single Point of Integration

Any integrated solution must be undertaken as a single logical entity. This entity can be seen as an extension of the application addressing any shortcomings of the application to present or receive business data in a guaranteed and consistent manner.

Separation of Concerns

All integration solutions must adhere to a well defined separation of concerns which do not unnecessarily bleed into each other.

Application Connectivity

All integration solutions (for both the sourcing and distribution of data) must isolate the application specific aspects regarding connectivity (namely the application API/protocols).

Application Data Constructs

Application specific data constructs containing the business data will be presented to and from the application. These constructs will exhibit syntax and semantics specific to the application integration. These must be resolved away from the application specifics to a common business form for sourced data and to the application specifics from a common business form for distributed data.

Common Data Operations

The common data (in terms of syntax and semantics) may require further "massage" and "value-add" in the form of filtering, routing, extension, mapping, aggregation and/or splitting. These operations are higher level data operations than that of the Application Data Constructs as the data is now operated on based on known business events.

Business Event Services

Business events exposed from source integration solutions or pushed to target integration solutions are done so via well defined contracts for event exchange.

Loose Coupling

All entities configured within the platform must be loosely coupled and exhibit loose dependencies in relation to each other. This is achieved through a defined public contract for entity interaction.

Tight Cohesion

Cohesion refers to the "degree of relatedness of services and/or operations" within an entity. It is important to have tight cohesion within the integration solutions to ensure that functionality is not unnecessarily distributed. Each solution should be self-contained and provide a defined contract for others wishing to use that service.

Standard Open Technologies

Each integration solution will require (a) specific knowledge of an application by the integration solution developers; and (b) specific knowledge of the business nature of the data being transported by the business analysts. Given these two requirements, the platform should not bog the developers or business analysts down with vendor specific or little known technologies beyond their remit. To this end it is imperative that the technologies be widely used and, where possible, industry standard. This should allow any developer to focus on the application specific integration issues and re-use industry knowledge without being subject to a steep learning curve for the infrastructure.

Standardisation through a Platform

The use of a common platform is born out of the requirement for application integration and business orchestration to be standard, modular, and re-useable.

Modularity

A common platform encourages modularity through encapsulation of related services and operations and exposing these as a well-defined contract between the solution and other entities.

Extensibility

The platform should provision for the extensibility of services and operations by providing hooks allowing designed extensions of services for entity specific operations.

Re-useability

A platform can provide generic implementations of components which can subsequently be re-applied across solutions. This leverages the domain knowledge and prior effort of the generic implementation allowing the developer to concentrate on the specifics of the component being created.

Inversion of Control

The platform provides a run-time implementation characterised by an “inversion of control”. This essentially means that an integration solution runs within the control of the platform.

Commoditised Solution

The implementation of services within a platform is not infinite. Their operation and use can be well defined, therefore, it is possible to determine common patterns of usage with regard to application integration projects. Once common patterns are identified it is possible to create templates and commoditise solutions to reduce the development effort.

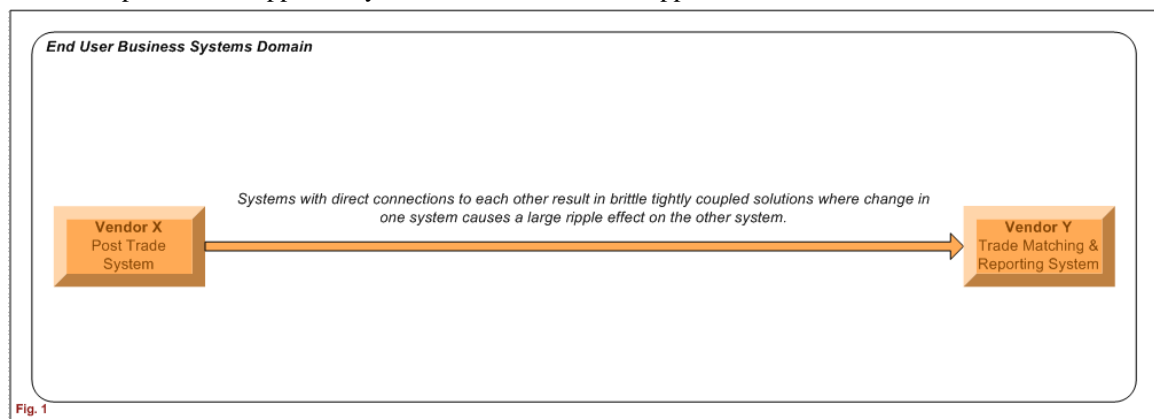
Chapter 3. Concepts

This chapter explains the architectural integration concepts underpinning the Iksan Enterprise Integration Platform approach and how these translate directly to Iksan design concepts.

Integration Concepts

True integration is about encompassing applications into wider scoped business flows as participants operating within their area of expertise. Referring back to the separation of concerns in the Architecture Strategies chapter we can see there are clearly defined integration layers each of which need to be addressed to achieve any desired solution.

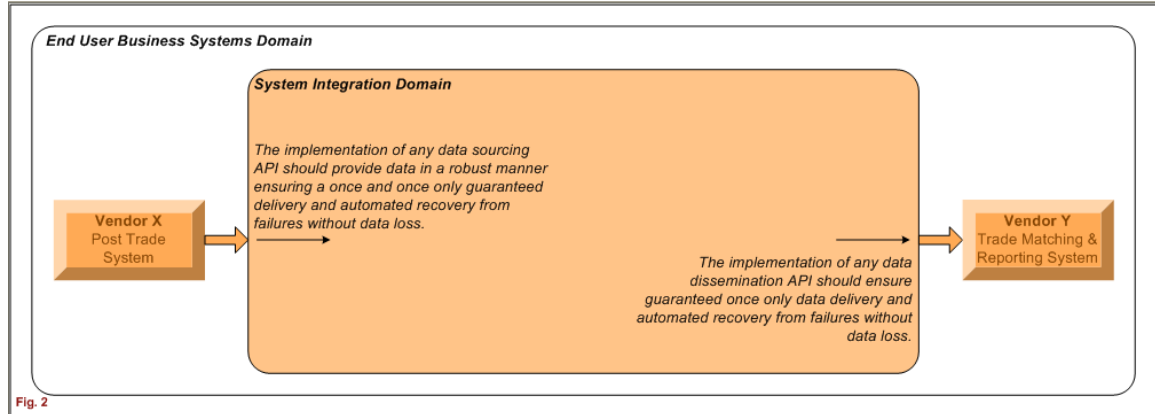
This section demonstrates these conceptual layers through the example of two applications exchanging data in a business flow. In fig.1, below, vendor X has trade confirmations (X's area of expertise) which need to go to vendor Y for reporting and matching (Y's area of expertise). Getting these applications to connect and exchange data directly is a possible solution, but not a very good. Direct (point to point) integration like this results in brittle solutions - a change in either vendor will have a direct impact on the other; and provides no opportunity to re-use should another application be introduced into the flow.



If we take this same example and start introducing the integration layers we start to see a very different picture.

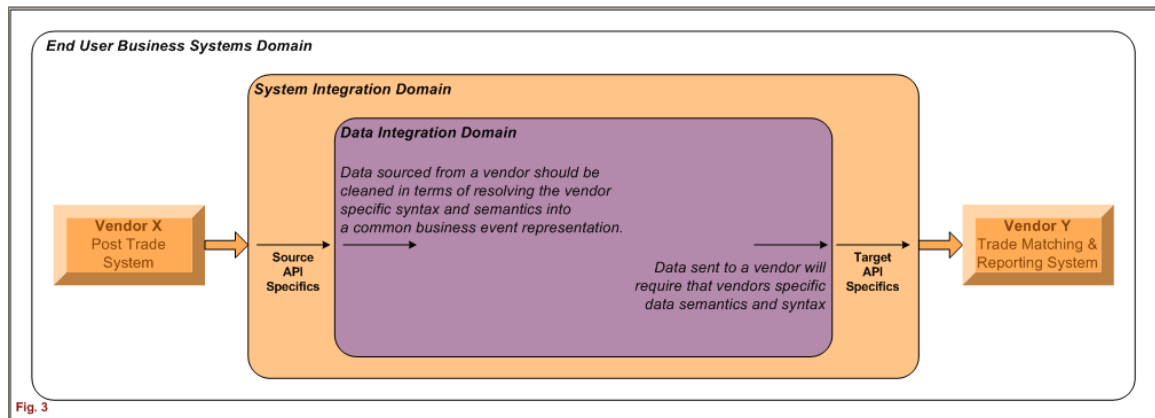
System Integration

System integration is about addressing the protocol requirements for accessing the data. In many cases this is about implementing a vendor specific API or accessing data through a more public protocol such as FTP or RDBMS (for database access).



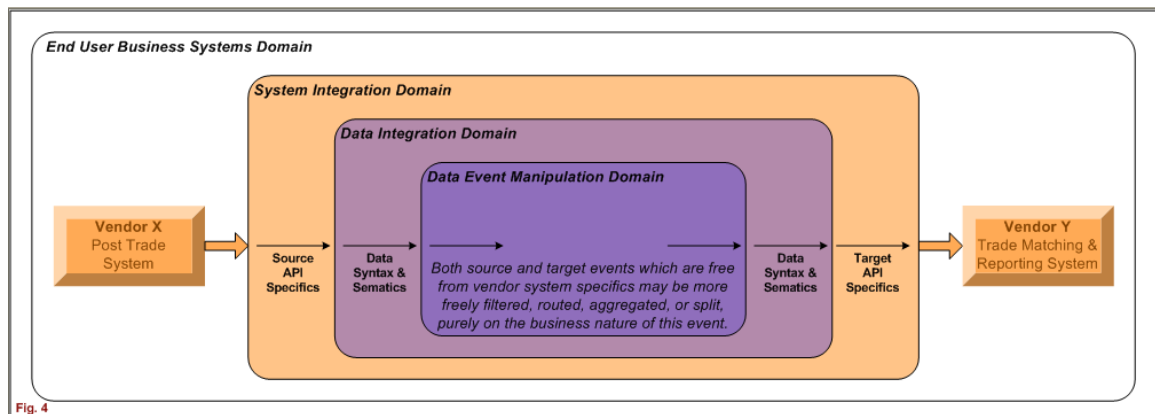
Data Integration

Data integration focuses on the data constructs provided from or presented to the application via the System Integration layer. Whatever form these data constructs take, this layer is about resolving the data construct syntax (format of the data) and semantics (meaning of the data).



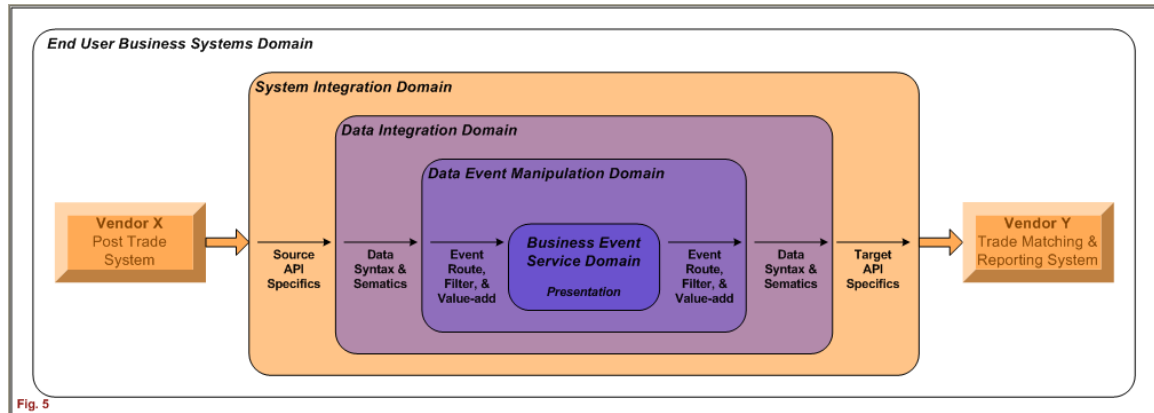
Event Integration

Event integration focuses on higher level data manipulation as cleaned business events. At this level, these data events are manipulated to implement basic business rules on known events. This can include routing, filtering, aggregation, splitting, etc.



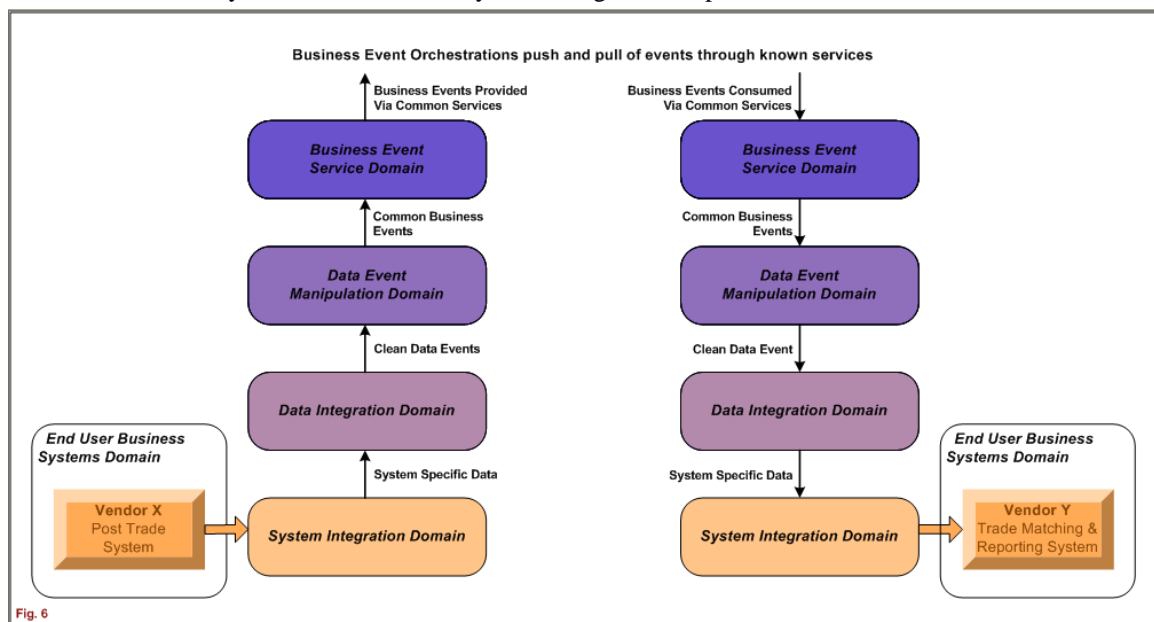
Event/Service Presentation

This layer is about presenting the polished business event through a well defined known service end point with which other systems can interact to receive and send business events.



Integration Layer View

Each of these four layers can be more clearly seen in fig.6 which presents these in a side view.



TODO - Ikasan Design

This section shows how these Integration Concepts have been translated into the design of the Ikasan platform.

Ikasan Design Overview

Diagram of all conceptual layers as Ikasan constructs.

Ikasan System Integration

Source: Application -> Data -> Payload -> Initiator Event Target: Endpoint Event -> Payload -> Data -> Application

Ikasan Data Integration

Source: Initiator Event -> Syntax Transformer -> Semantic Transformer -> Event Target: Event -> Semantic Transformer -> syntax Transformer -> endpoint

Ikasan Event Integration

Source/Target: Event -> Aggregator/Router/Splitter -> Event

Ikasan Event Presentation

Source/Target: Event -> Endpoint

Index