
# ikasan

Est. 2005

# Open Source Enterprise Integration Platform

June 2020

Info@ikasan.org

# Introduction

- The Problem Domain
- Architecture Strategies
- Translating the Architecture
- Ikasan Services
- Business Application Integration
- Technology Stack
- Ikasan Releases
- Resources

# The Problem Domain

## Domain

Enterprise Application Integration (EAI) space

- integration of business systems in a landscape of interwoven and often complex finance processes

## Issue

EAI is complex, costly, & not the primary business concern

- greater adoption of "Best of Breed" specialist applications
- business applications distributed across disparate platforms
- business data distributed across isolated silos
- legacy data repositories
- data duplication and integrity issues
- no clear business data owner
- exponential integration requirements
- greater complexity of business demand on data orchestration
- EAI is more than simply connecting applications

## Goal

To provide simple, robust, configurable commoditised solutions

- expose business artifacts whilst isolating the integration specifics

# Architecture Strategies

## Simplicity

- must not require in-depth proprietary knowledge
- avoidance of overly complex or heavyweight frameworks
- highly testable simple constructs

## Commonality

- reusable interchangeable constructs
- standard contracts of interaction
- simple repeatable implementation steps

## Adaptability

- ability to support any type of business entity
- ability to integrate any type of Enterprise Information System
- without breaking the first two strategies of Simplicity & Commonality

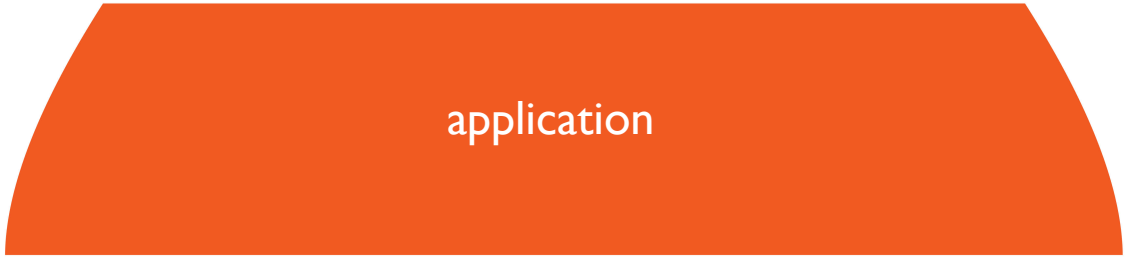
## Robust & Guaranteed operation

- maintain data integrity in business delivery and failure scenarios
- require minimal manual intervention i.e. failure / automated recovery
- clear contract definition and separation of concerns

# Architecture Strategies

clear contract definition and separation of concerns

- application concerns



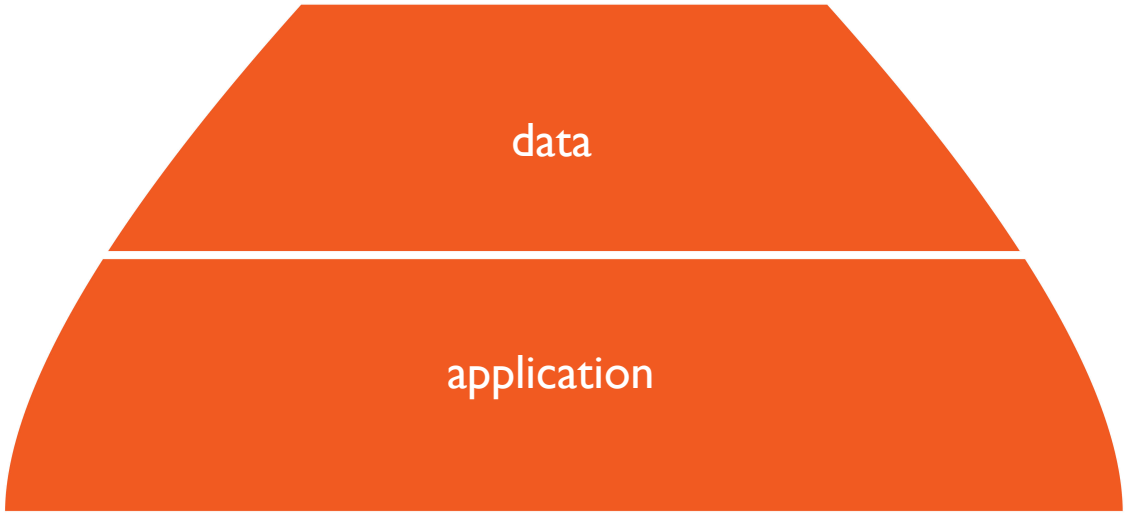

application

performant, robust & guaranteed operation

# Architecture Strategies

clear contract definition and separation of concerns

- application concerns
- data concerns




data

presentation of standard, meaningful business data

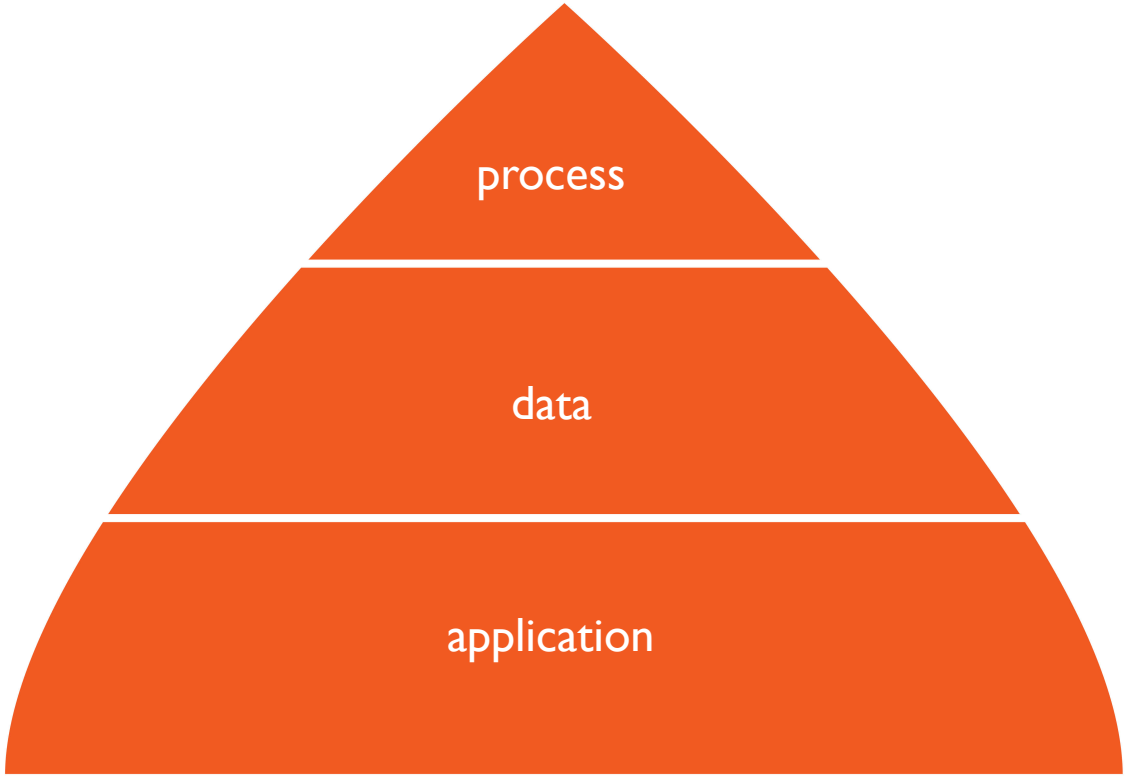
application

performant, robust & guaranteed operation

# Architecture Strategies

clear contract definition and separation of concerns

- application concerns
- data concerns
- business process concerns




process

business event & entity orchestration (STP is the goal)

data

presentation of standard, meaningful business data

application

performant, robust & guaranteed operation

# Architecture Strategies

## Loose coupling

- integrations should only require knowledge of each other through exchange of business artifacts

## Tight cohesion

- an integration's specifics must not bleed out unnecessarily

## High visibility

- business data tracking
- fail scenarios
- operational audit

## Single logical point of integration

- geographically agnostic / cross platform support

## Open standards aligned with proven design patterns

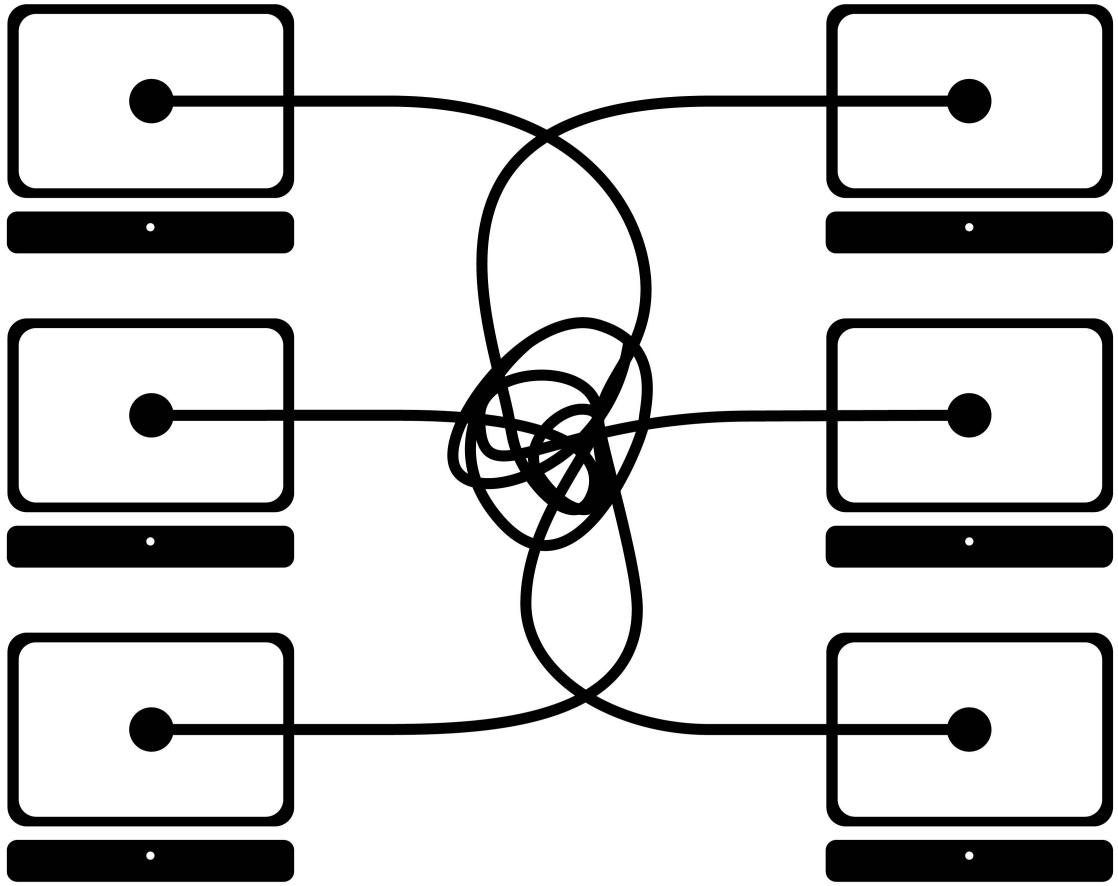
- adoption of industry standard Enterprise Design Patterns avoidance of vendor lock-in

## Ensure traceability from architecture through to implementation

- all too often architecture fails to translate into the real world
- the right architecture can reduce complexity and drive down cost of change and support

# Translating the Architecture

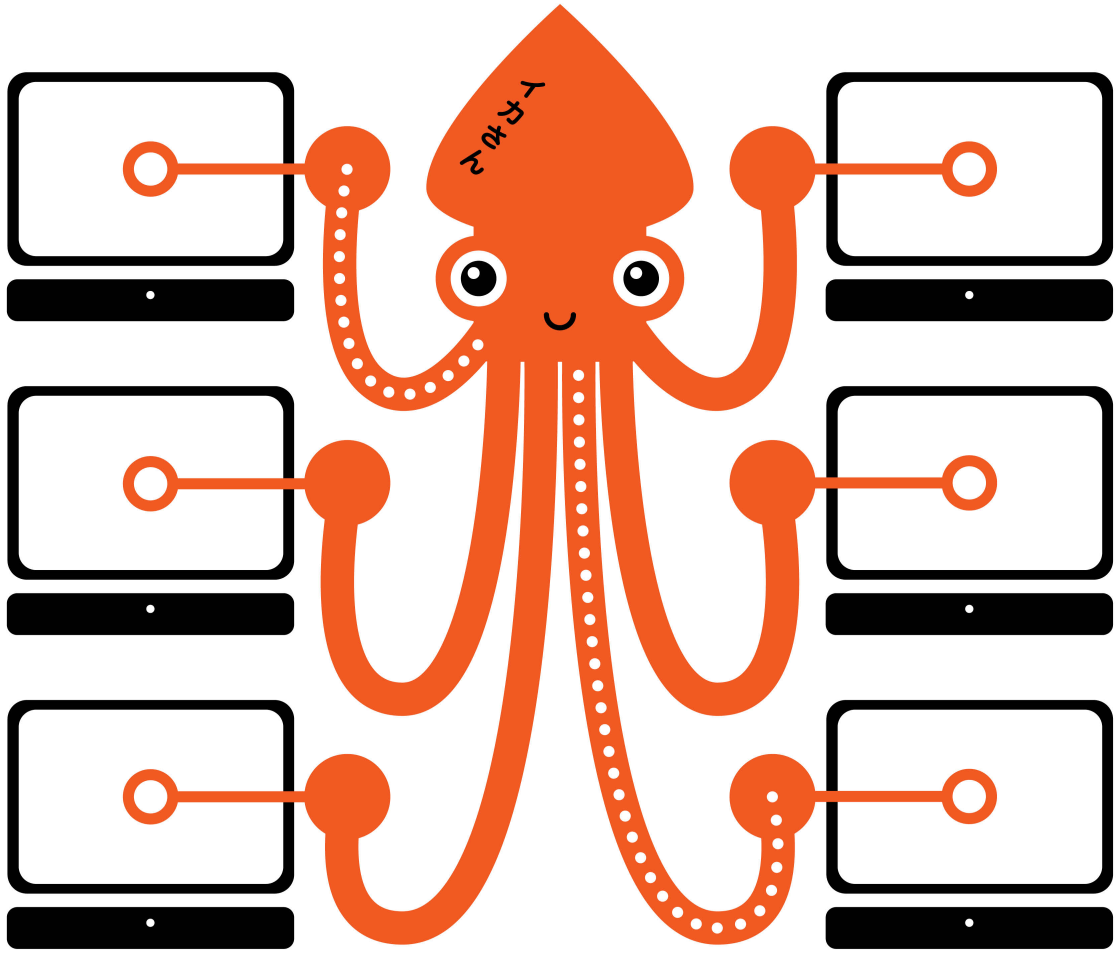
Commonly agreed that "spaghetti integration" is bad



- integration can be exponentially complex
- bleed out of APIs, data syntax and business semantics
- ripple effect of change is massive

# Translating the Architecture

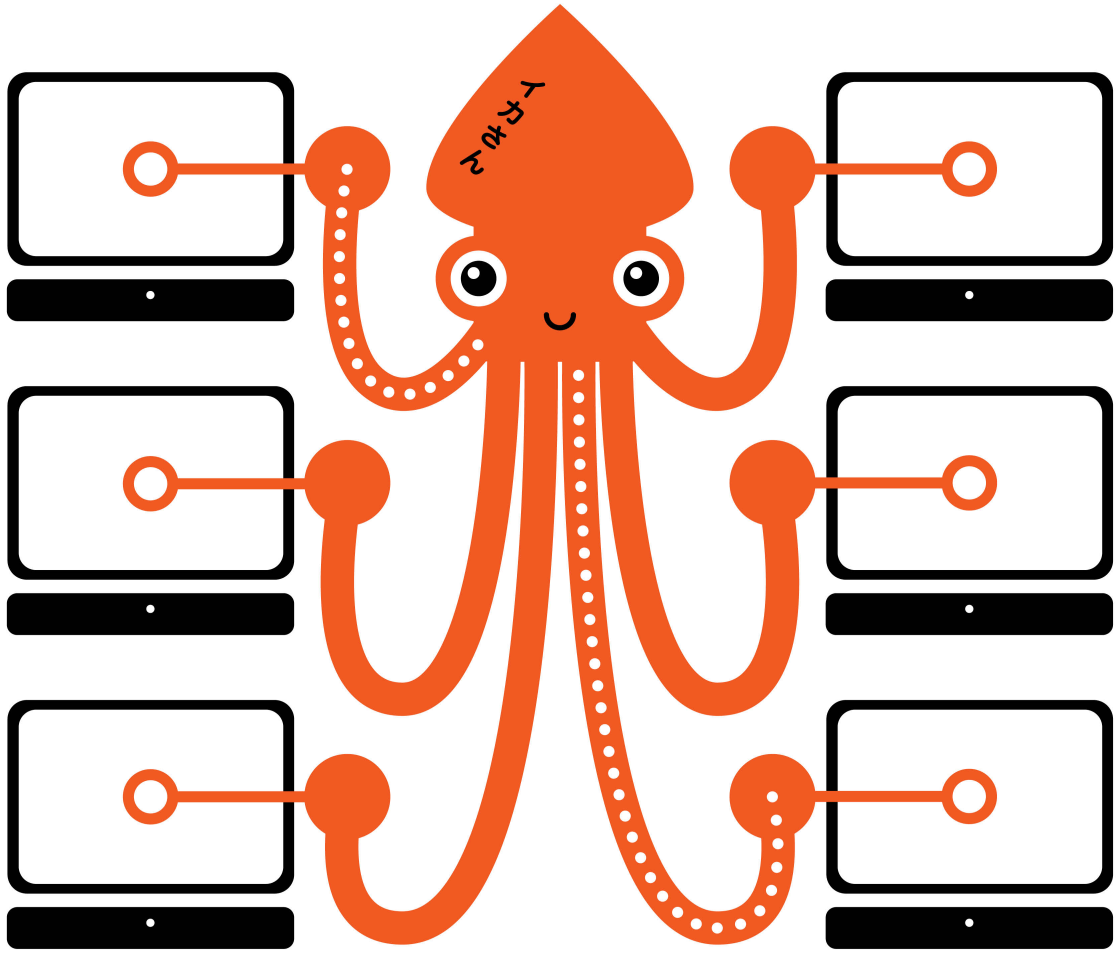
## Standard Enterprise Application Integration (EAI) approach



- provision of an Event/Service backbone
- single point of integration for EIS business flow
- ripple effect of change is localized
  *(assuming best practice)*

# Translating the Architecture

## Standard Enterprise Application Integration (EAI) approach



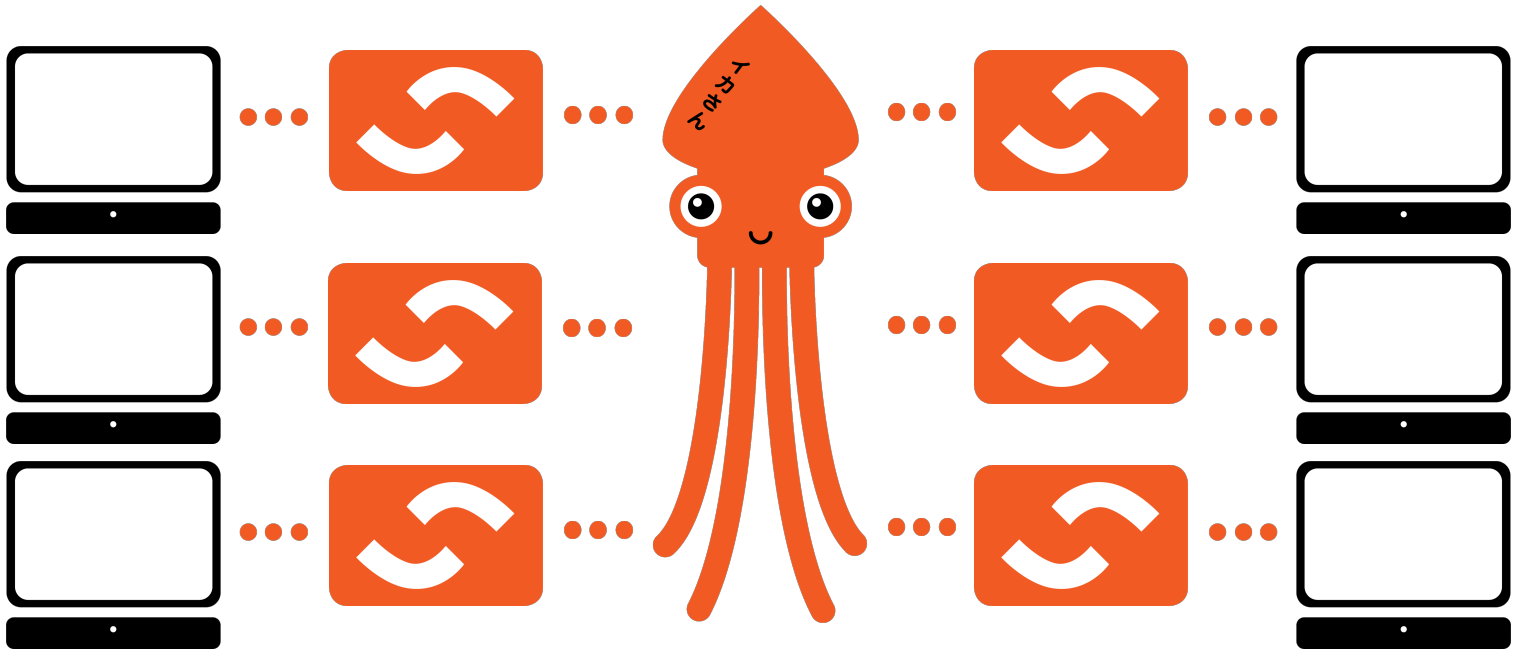
### First Steps

- understand your end to end business streams
- understand the participating domains
- define and model your enterprise domain
  - model entities as data & behaviour
  - anaemic entities (data only) decay over time
- leverage existing domain knowledge
  - industry standards likely exist
  - Enterprise Integration Design Patterns
  - Enterprise Application Design Patterns
- identify the integration contract touch points
- define IkasanESB Integration points

# Translating the Architecture

## IkasanESB Integration Modules

- provide a grouping of like business application operations
- single logical integration point
- supports sourcing, distribution, and bi-directional event data
- built and run as a self-contained service



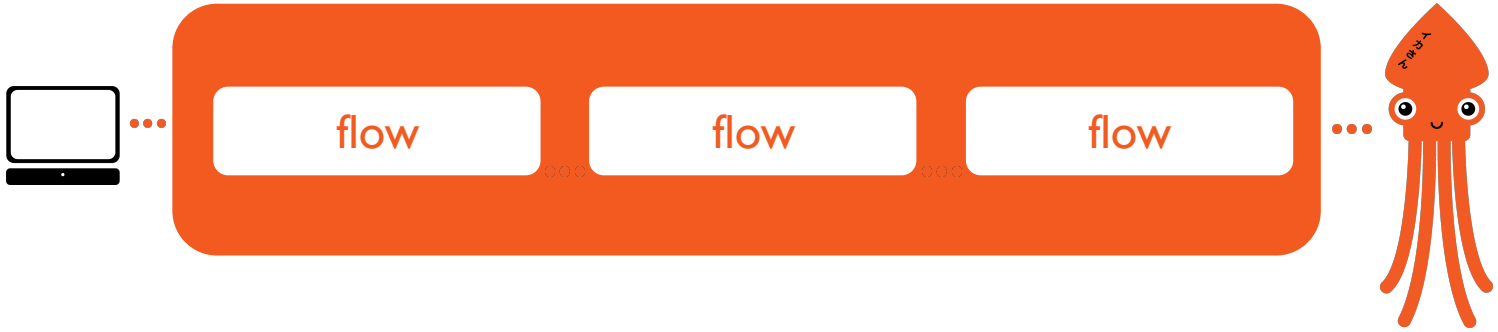
Key Strategies

- ✓ single point integration
- ✓ loose coupling
- ✓ variety of standard contracts
- separation of concerns
- tight cohesion
- standard design patterns
- interchangeable constructs
- highly testable constructs
- data integrity
- management & control
- monitor & alerting
- minimal manual intervention
- support any protocol/entity
- business event tracking
- operational audit

# Translating the Architecture

## IkasanESB Flows

- flows belong to an Integration Module
- provide cohesive, atomic operations on business artifacts
- multiple flows can be chained to isolate concerns
- standard event container allows any data type to be transported



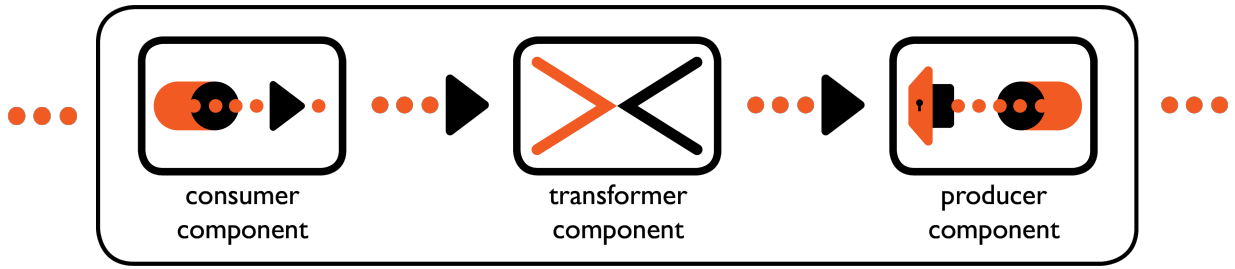

## Key Strategies

✓ single point integration
✓ loose coupling
✓ variety of standard contracts
✓ separation of concerns
✓ tight cohesion
✓ standard design patterns
• interchangeable constructs
• highly testable constructs
• data integrity
• management & control
• monitor & alerting
• minimal manual intervention
• support any protocol/entity
• business event tracking
• operational audit

# Translating the Architecture

## IkasanESB Flow Components

- belong to a Flow
- POJO implementations provide independent, interchangeable operations
- isolated white box testing and chained black box testing
- unfettered event transport without overhead of casting or serialization



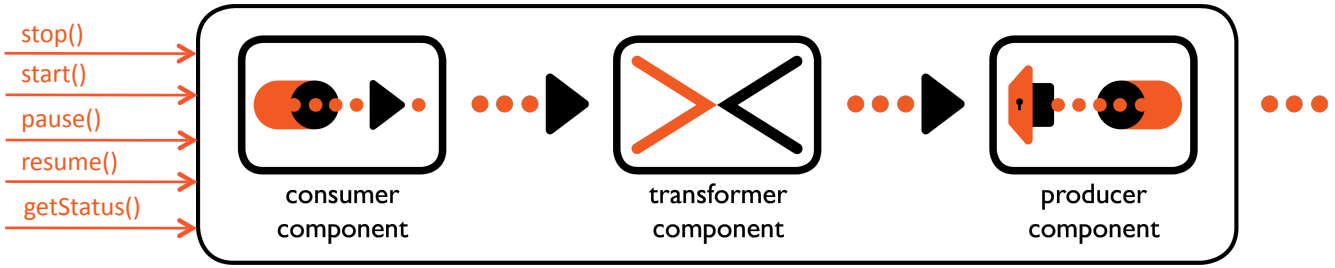

## Key Strategies

- ✓ single point integration
- ✓ loose coupling
- ✓ variety of standard contracts
- ✓ separation of concerns
- ✓ tight cohesion
- ✓ standard design patterns
- ✓ interchangeable constructs
- ✓ highly testable constructs
- ✓ data integrity
- • management & control
- • monitor & alerting
- • minimal manual intervention
- • support any protocol/entity
- • business event tracking
- • operational audit

# Translating the Architecture

## IkasanESB Flow Contracts

- management API for business flow control (start, stop, pause, resume)
- runtime status API for real-time health (running, stopped, recovering, error)
- monitor API for pluggable health notification as real-time alerting
- recovery manager for configurable exception handling & automated recovery



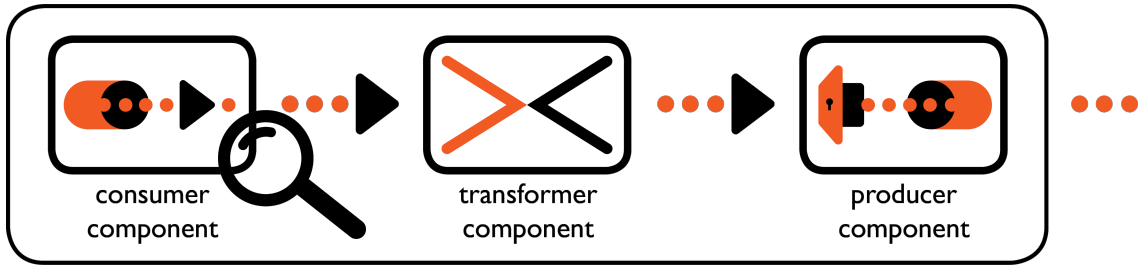

Key Strategies

- ✓ single point integration
- ✓ loose coupling
- ✓ variety of standard contracts
- ✓ separation of concerns
- ✓ tight cohesion
- ✓ standard design patterns
- ✓ interchangeable constructs
- ✓ highly testable constructs
- ✓ data integrity
- ✓ management & control
- ✓ monitor & alerting
- ✓ minimal manual intervention
- • support any protocol/entity
- • business event tracking
- • operational audit

# Translating the Architecture

## IkasanESB Component Types

- consumers are always the starting component for any flow
- consumers can support any type of protocol
- extend guaranteed integrity to non-guaranteed EIS sources/targets
- no proprietary knowledge required – only EIS API
- API adaptors are pluggable and re-useable components
- API support can be 3rd party off-the-shelf or custom developed



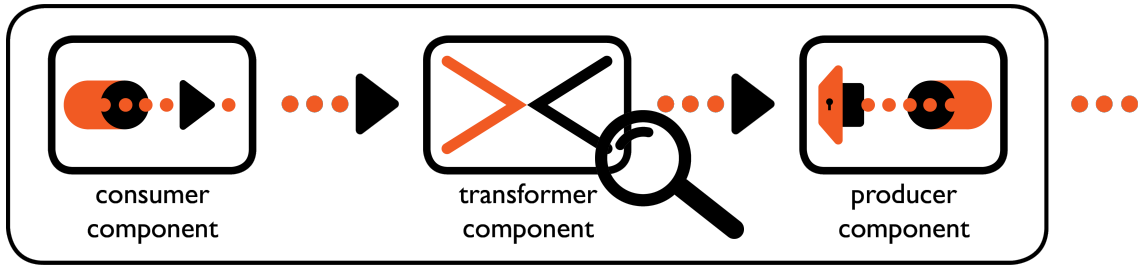

Key Strategies

- ✓ single point integration
- ✓ loose coupling
- ✓ variety of standard contracts
- ✓ separation of concerns
- ✓ tight cohesion
- ✓ standard design patterns
- ✓ interchangeable constructs
- ✓ highly testable constructs
- ✓ data integrity
- ✓ management & control
- ✓ monitor & alerting
- ✓ minimal manual intervention
- ✓ support any protocol/entity
- • business event tracking
- • operational audit

# Translating the Architecture

## IkasanESB Component Types

- any number of components can operate on data following a consumer
  - converter
  - translator
  - splitter
  - filter
  - sequencer
  - broker
  - single-recipient router
  - multi-recipient router
  - producer



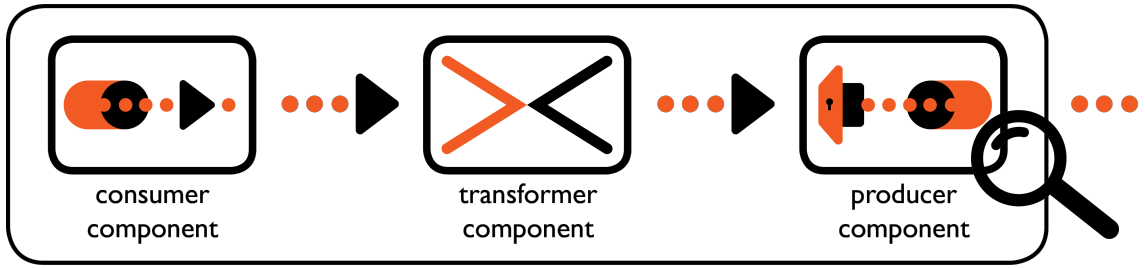

Key Strategies

- ✓ single point integration
- ✓ loose coupling
- ✓ variety of standard contracts
- ✓ separation of concerns
- ✓ tight cohesion
- ✓ standard design patterns
- ✓ interchangeable constructs
- ✓ highly testable constructs
- ✓ data integrity
- ✓ management & control
- ✓ monitor & alerting
- ✓ minimal manual intervention
- ✓ support any protocol/entity
- • business event tracking
- • operational audit

# Translating the Architecture

## IkasanESB Component Types

- producer components always end a flow
- several producers may exist in the flow if the flow has branched routes



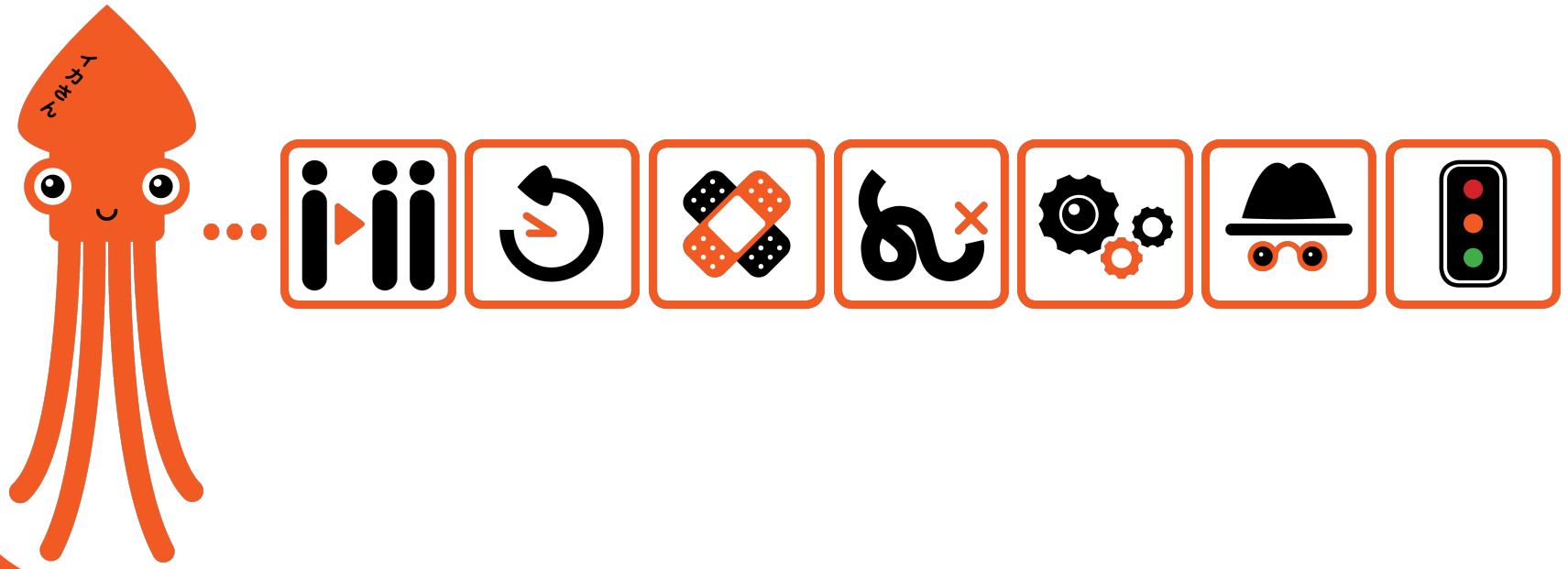

Key Strategies

- ✓ single point integration
- ✓ loose coupling
- ✓ variety of standard contracts
- ✓ separation of concerns
- ✓ tight cohesion
- ✓ standard design patterns
- ✓ interchangeable constructs
- ✓ highly testable constructs
- ✓ data integrity
- ✓ management & control
- ✓ monitor & alerting
- ✓ minimal manual intervention
- ✓ support any protocol/entity
- • business event tracking
- • operational audit

# Translating the Architecture

IkasanESB Dashboard

- single pane view of the entire IkasanESB Integration Module estate
- management and administration for all services including
  - user/group principals, roles, and policies
  - configuration of Integration Modules, flows, and components
  - sub-second searches of data events across the entire estate
  - errors, exclusions, resubmissions, and event recording and replay
- typically used by IT support and business teams



Key Strategies

- ✓ single point integration
- ✓ loose coupling
- ✓ variety of standard contracts
- ✓ separation of concerns
- ✓ tight cohesion
- ✓ standard design patterns
- ✓ interchangeable constructs
- ✓ highly testable constructs
- ✓ data integrity
- ✓ management & control
- ✓ monitor & alerting
- ✓ minimal manual intervention
- ✓ support any protocol/entity
- ✓ business event tracking
- ✓ operational audit

# Ikasan Services

## Ikasan Standard Services

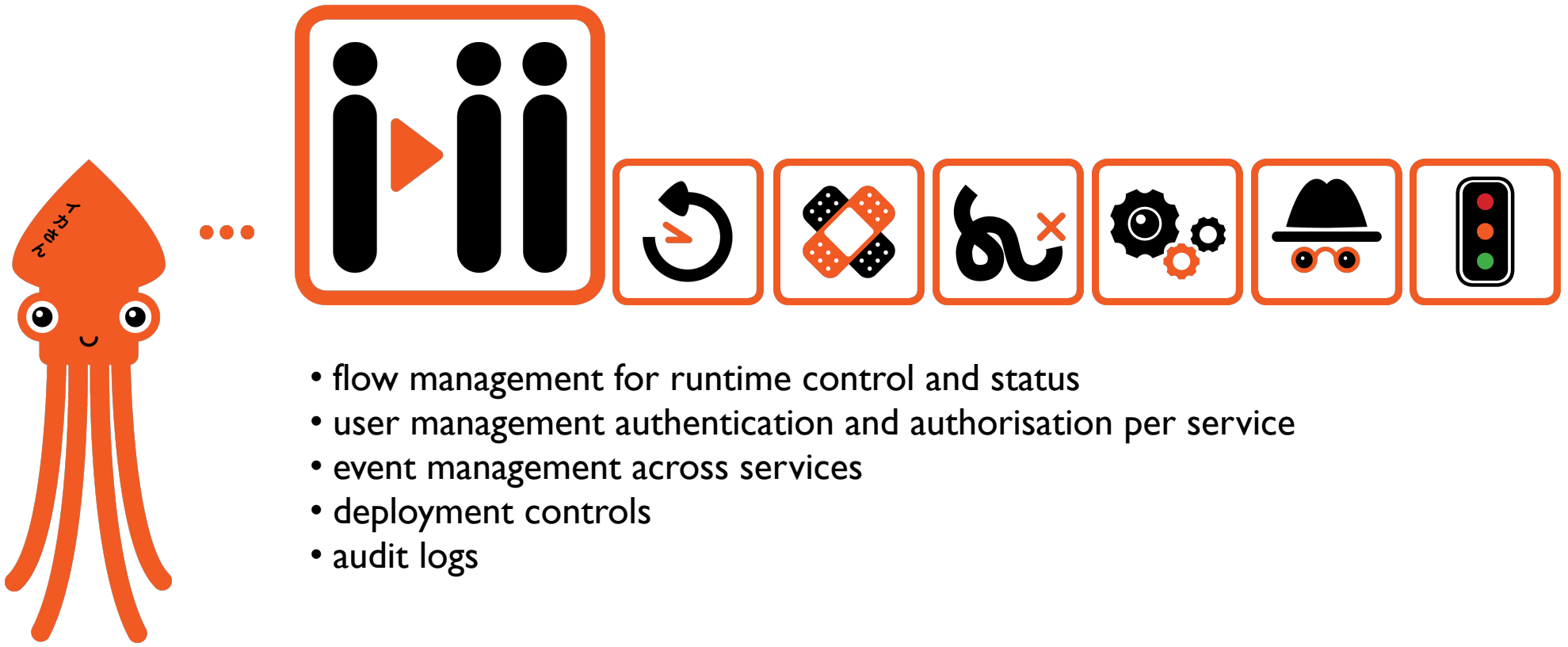
- standard services are automatically included with each Integration Module
- service contracts are pluggable, so replacing default implementation with custom is possible
- standard services are summarised below

- Management Service
- Record & Replay Service
- Hospital Services
- Mapping Service
- Configuration Service
- Wiretap Service
- Monitoring Service

# Ikasan Services

## Management Service

- management service endpoints are available across all Ikasan artefacts
- Ikasan Dashboard provides a single pane view of all Integration Modules and management therein
- REST API service access for automation of calls to services

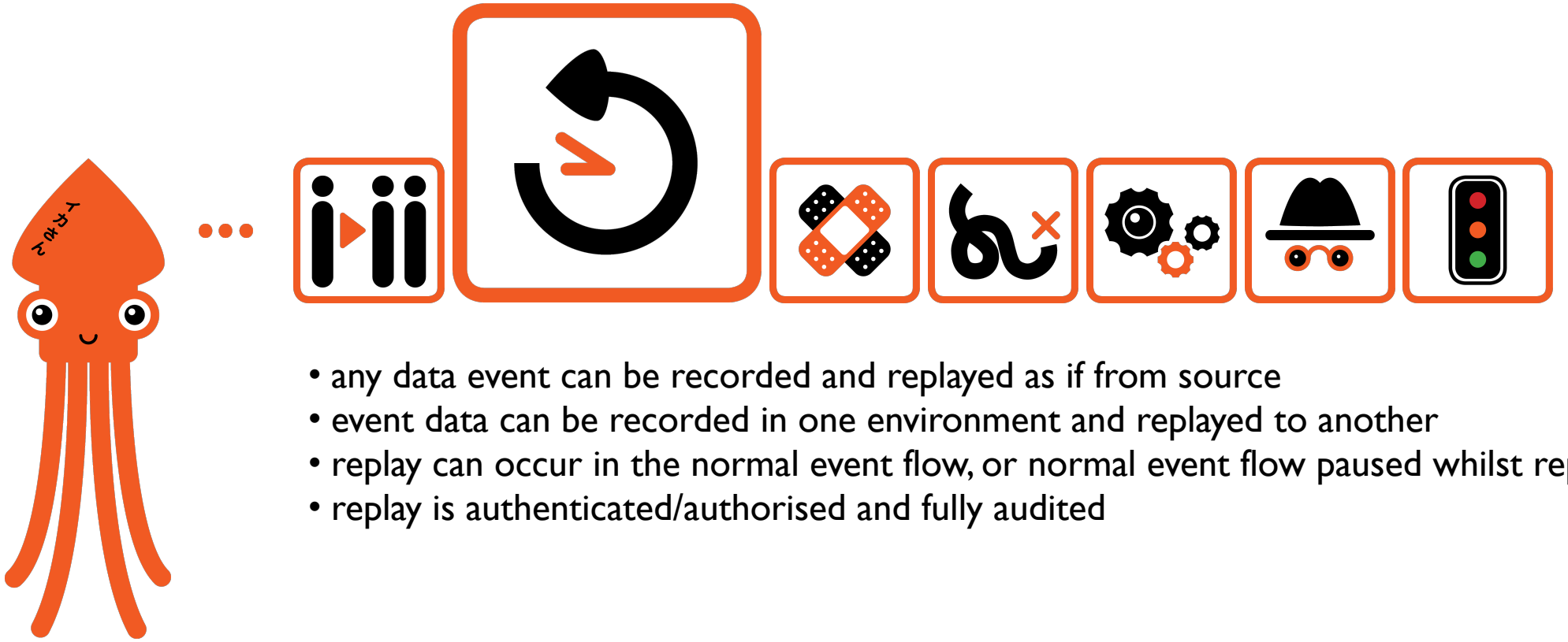


- flow management for runtime control and status
- user management authentication and authorisation per service
- event management across services
- deployment controls
- audit logs

# Ikasan Services

## Record & Replay Service

- supports event recording per flow
- events can be replayed to this or another Ikasan instance
- invaluable for EIS business system crash/recovery or production data replay to test environments

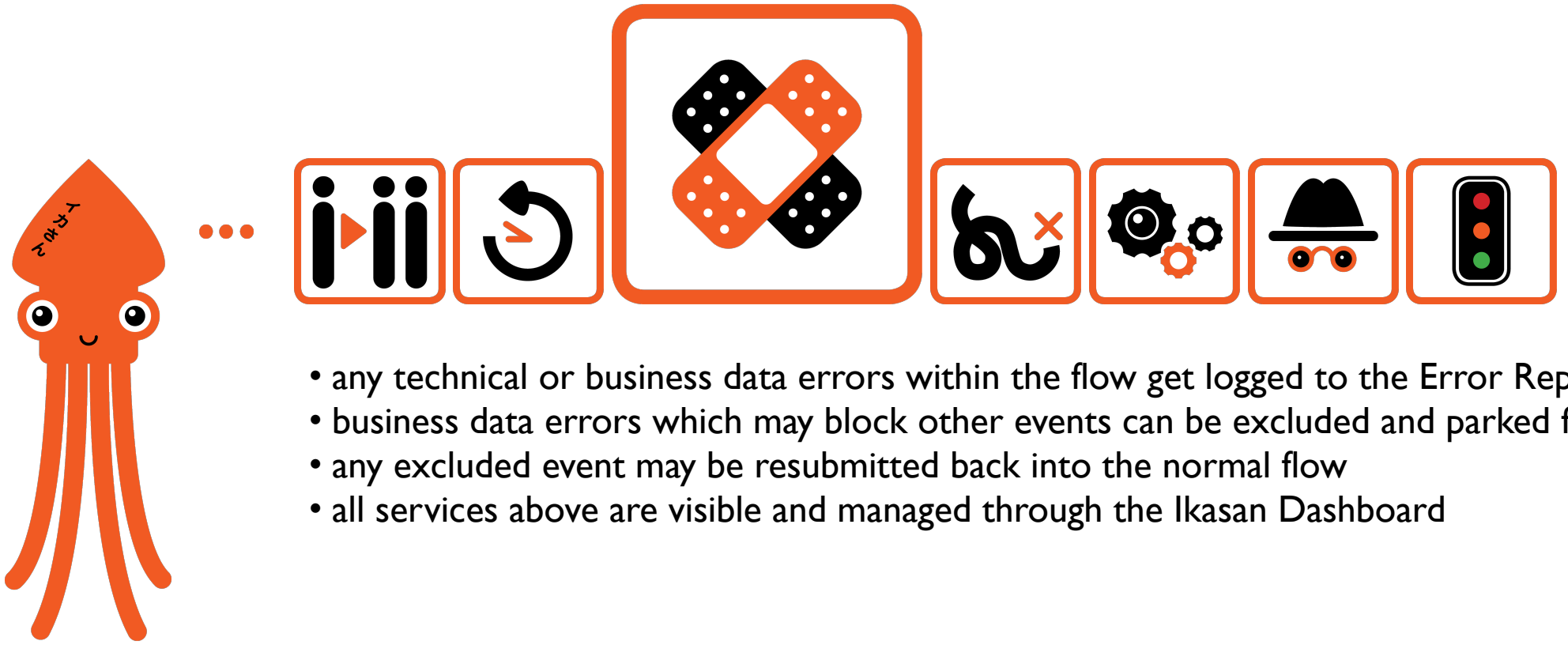


- any data event can be recorded and replayed as if from source
- event data can be recorded in one environment and replayed to another
- replay can occur in the normal event flow, or normal event flow paused whilst replay is in progress
- replay is authenticated/authorised and fully audited

# Ikasan Services

## Hospital Services

- error reporting service provides visibility of all errors
- exclusions service allows individual bad events to be removed from the flow
- resubmission service supports the resubmission of previously error events back into the flow

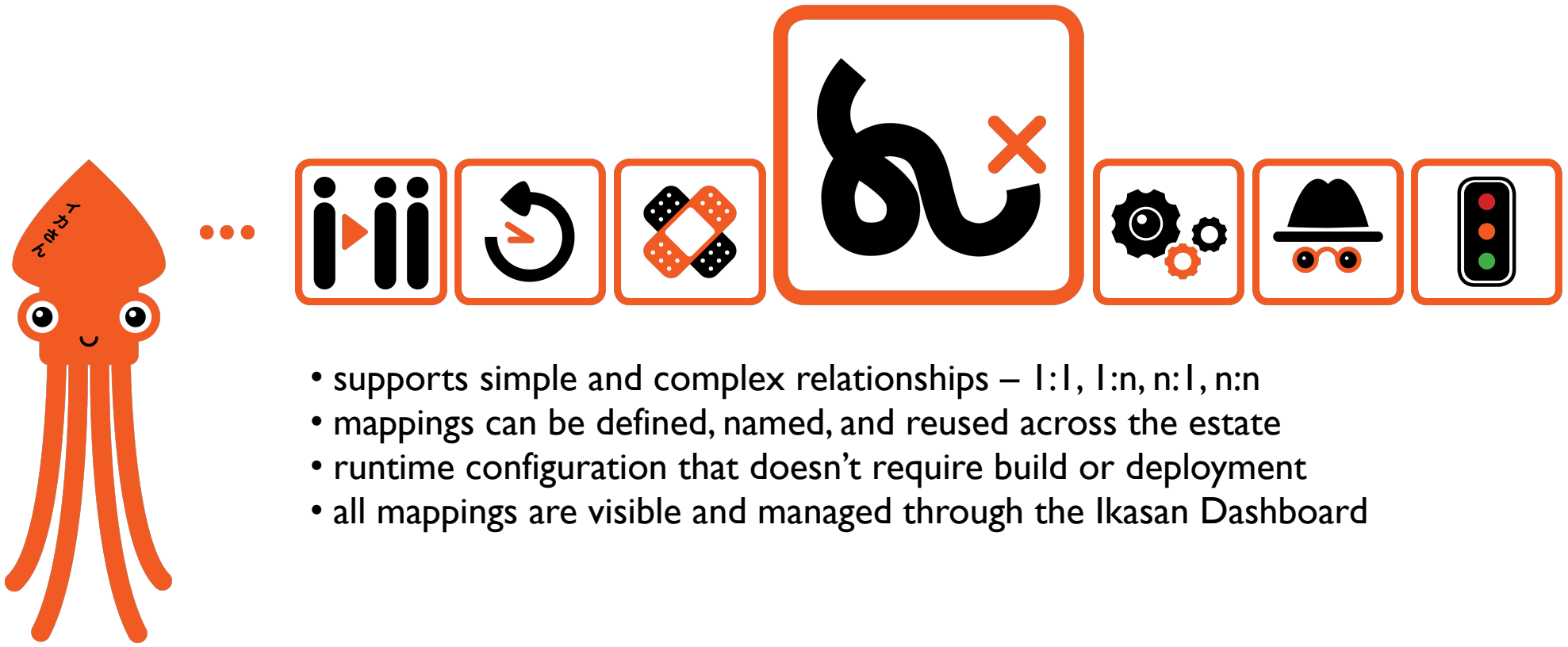


- any technical or business data errors within the flow get logged to the Error Reporting Service
- business data errors which may block other events can be excluded and parked for user attention
- any excluded event may be resubmitted back into the normal flow
- all services above are visible and managed through the Ikasan Dashboard

# Ikasan Services

## Mapping Service

- provides event attribute mapping as part of data transformation

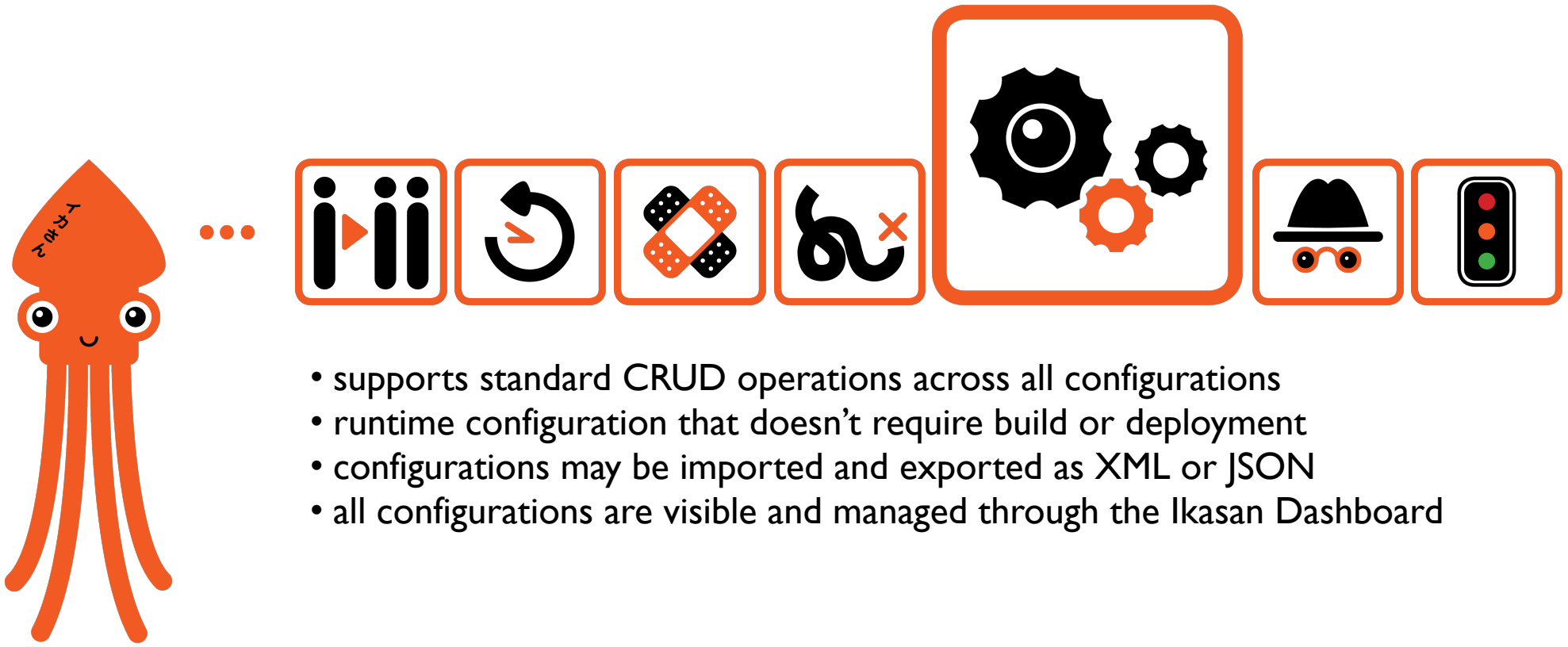


- supports simple and complex relationships – 1:1, 1:n, n:1, n:n
- mappings can be defined, named, and reused across the estate
- runtime configuration that doesn't require build or deployment
- all mappings are visible and managed through the Ikasan Dashboard

# Ikasan Services

## Configuration Service

- allows runtime configuration for all Ikasan artefacts i.e. components, flows, monitors

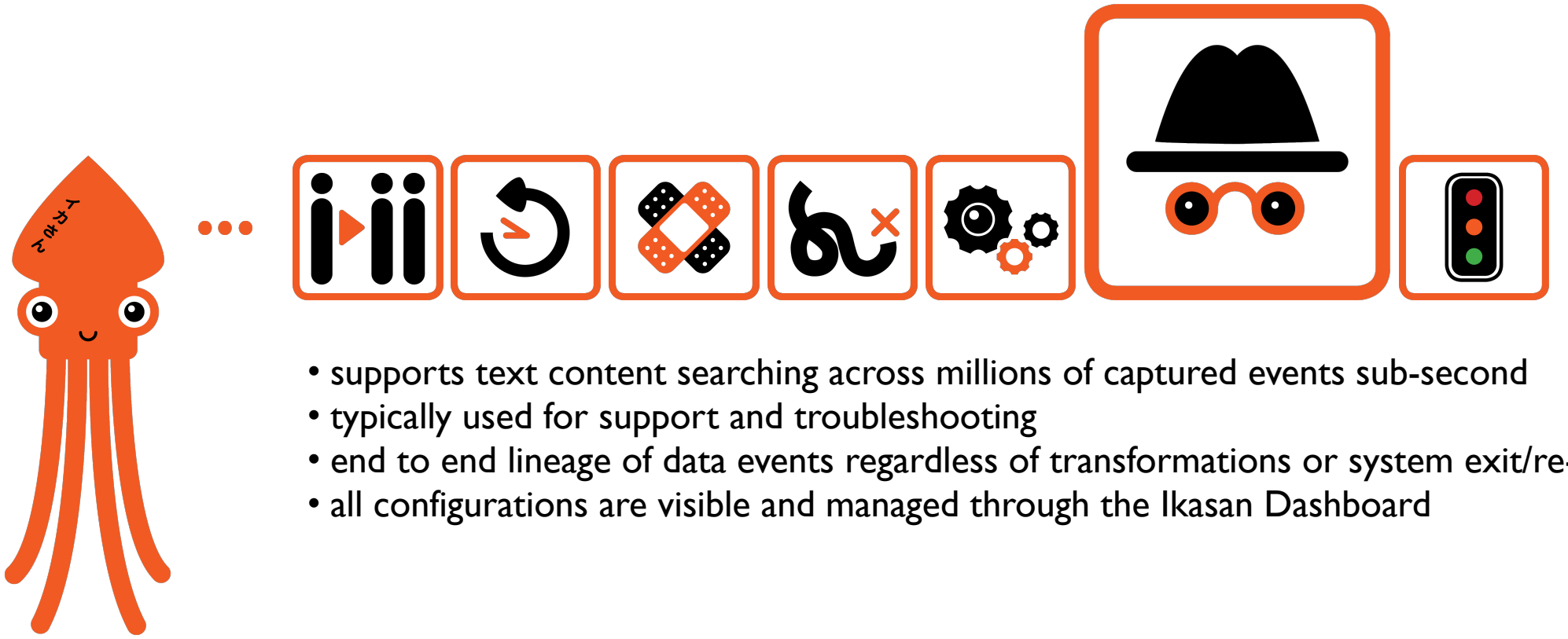


- supports standard CRUD operations across all configurations
- runtime configuration that doesn't require build or deployment
- configurations may be imported and exported as XML or JSON
- all configurations are visible and managed through the Ikasan Dashboard

# Ikasan Services

## Wiretap Service

- dynamic capture of real-time in-flight events at any point within the platform

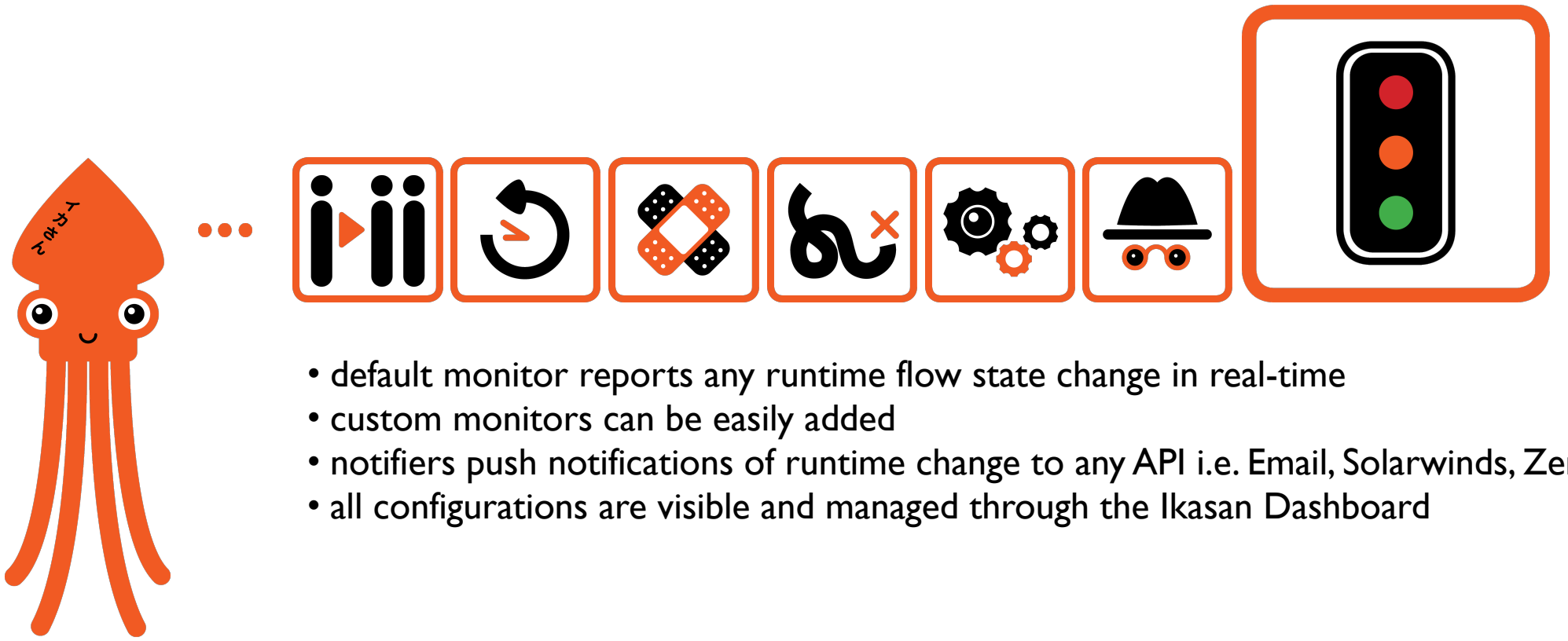


- supports text content searching across millions of captured events sub-second
- typically used for support and troubleshooting
- end to end lineage of data events regardless of transformations or system exit/re-entry
- all configurations are visible and managed through the Ikasan Dashboard

# Ikasan Services

## Monitor Service

- capture any state change in runtime flows i.e. stopped, paused, running, recovering, error
- notifiers connect to external Enterprise Monitoring tools to report state changes



- default monitor reports any runtime flow state change in real-time
- custom monitors can be easily added
- notifiers push notifications of runtime change to any API i.e. Email, Solarwinds, Zenoss, HTTP
- all configurations are visible and managed through the Ikasan Dashboard

# Business Application Integration

## Post Trade Execution

- Bloomberg
  - Trade Feed
  - Consolidated Message Feed
  - Derivative Settlement Feed
  - Daily Trade Reporting

- PATS
- Espeed
- TradeWeb
- ION, Anvil, & Fidessa

## Positions

- ION
- Bloomberg
- Intellimatch

## Price Marks

- ION
- Bloomberg

# Business Application Integration

## Trade Matching, Reporting & Reconciliation

- TRAX
- Bloomberg Daily Trade Reporting

## Sales Commission Reporting

- FIOS

## Reference Data

- Bloomberg Data License Per Security; Data License Back Office
- MACE Convertible Bonds
- Swap Monitor Financial Calendar
- Fidessa ETP and ATP
- GoldenSource

## Market Data

- Markit CDS Service
- Markit iBoxx Benchmark Indices
- Asset Control
- Bank of America Merrill Lynch Indices

# Business Application Integration

## Settlements & Operations

- SWIFT Alliance
- GlobOp

## Ratings

- Moodys
- Standard & Poors
- Fitch

## Compliance

- Thomson Reuters TransWatch Securities Data
- B-Next

## Corporate Actions

- FTI Corporate Action & Securities Universe

# Technology Stack

## Development Stack

- Core Technologies
  - Java 1.8+
  - Java 11 supported by Ikasan 3.1.0
  - SpringBoot

- Test Frameworks
  - JUnit
  - JMock
  - SpringBootTest
  - Ikasan Flow Test Harness

- Scaffolding
  - source management, peer review, acceptance management – GitHub
  - requirements and issues management – Jira
  - project build management – Maven
  - project artefact management – Sonotype Nexus
  - continuous integration – Travis

# Technology Stack

## Ikasan Runtime Stack

- IkasanESB Application
  - Ikasaneip-3.0.0 – engine underpinning all operations and services

- IkasanESB Optional Enterprise Modules
  - Ikasan-test-component – endpoint components for test event generation

- Persistence
  - H2 shipped as recommended production supported version
  - any Enterprise relational database (MySQL, SQLServer, Sybase, etc)

- Ikasan Dashboard
  - single pane view for all runtime Ikasan Integration Modules

- High Availability
  - Ikasan is tested/ratified to run in Docker containers as part of a Kubernetes managed platform

# Technology Stack

## Off-the-shelf Integration

- JMS (HornetQ, JBossA-MQ, ActiveMQ, IBM MQ, WebLogic)
- FTP (transactional & chunking support)
- SFTP (transactional & chunking support)
- RDBMS
- MQ Client
- FIX
- SMTP
- HTTP(s)
- MongoDB
- Kxdb
- HDFS & Parquet

## Custom Component Development

- very simple to create custom integrations specific to an upstream or downstream API
- all components adhere to standard contracts
- developed as standard Java 1.8+ (Java 11 for Ikasan 3.1.0 onwards)

# Technology Stack

## Flow Non-Functional Features

- High performance single threaded throughput
  - core engine benchmarked at 2 million events per second

- Supports full transaction semantics
  - non-transactional resources
  - local transactional resources
  - 2-phase transactional resources
  - last resource commit optimisation (LRCO)

- Configurable automated runtime recovery from technical and data failures
  - rollback operations and stop
  - rollback operations and retry
  - rollback and exclude bad data event
  - report the issue and continue

# Technology Stack

## Development times are a function of either complexity or volume

- functionally complex Integration Modules will naturally take longer to implement and test than simple ones
- a simple Integration Module containing 10 flows will take longer than one with 2 flows

## Ikasan Integration Module implementation guide

*The following are provided as indicative efforts only*

- low complexity based on Maven archetypes
  - 1 day for a simple Integration module containing a 1-2 flows consisting of 3-4 components each
  - 3 days for a simple integration module containing 3 flows consisting of 5 components each

- moderate complexity based on Maven archetypes
  - 1 – 10 days for an Integration module containing a 3-5 flows each of which has 5-10 components

- high complexity
  - 15 – 20 days for a complex module containing 6-8 flows each of which has 10-15 components
  - 20 – 30 days for a complex module containing 8+ flows each of which has 15+ components

# Ikasan Releases

## Bug Fix Releases

- ad-hoc for emergencies
- monthly

## Minor Releases

- quarterly

## Major Releases

- half yearly

## Release Compatibility

- all Ikasan releases are compatible with previous releases from Ikasan v2.0.0 onwards
- any updates / migrations within the Ikasan private persistence are automated on deployment
- multiple versions of Ikasan can run simultaneously

# Resources

## Web Site

- http://www.ikasan.org

## GitHub

- https://github.com/ikasanEIP/ikasan

## Jira / Confluence

- https://ikasan.atlassian.net/projects/IKASAN

## Developer Guides

- https://github.com/ikasanEIP/ikasan/blob/master/ikasaneip/developer/docs/StandaloneDeveloperGuide.md

## Contact

- info@ikasan.org