Open Source Enterprise Integration Platform

Jeff Mitchell & Jun Suetake
ikasan.org
July 2014

Introduction

- The Problem Domain
- Architecture Strategies
- Translating the Architecture
- Technology Stack
- Features
- Development Lifecycle
- Roadmap
- Summary

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

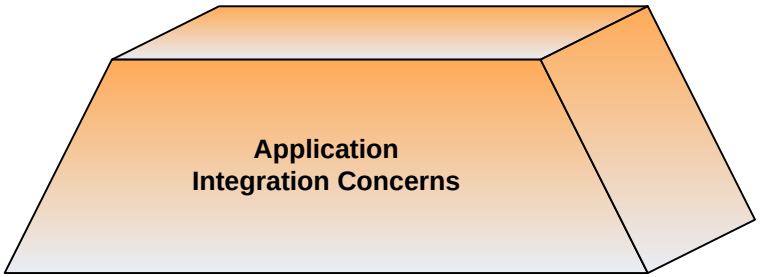
- Simplicity
  - Must not require in-depth proprietary knowledge
  - Avoidance of overly complex or heavyweight frameworks
  - Highly testable simple constructs
- Commonality
  - Reusable interchangeable constructs
  - Standard contracts of interaction
  - Simple repeatable implementation steps
- Adaptability
  - 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

Architecture Strategies

- Clear contract definition and separation of concerns

Architecture Strategies

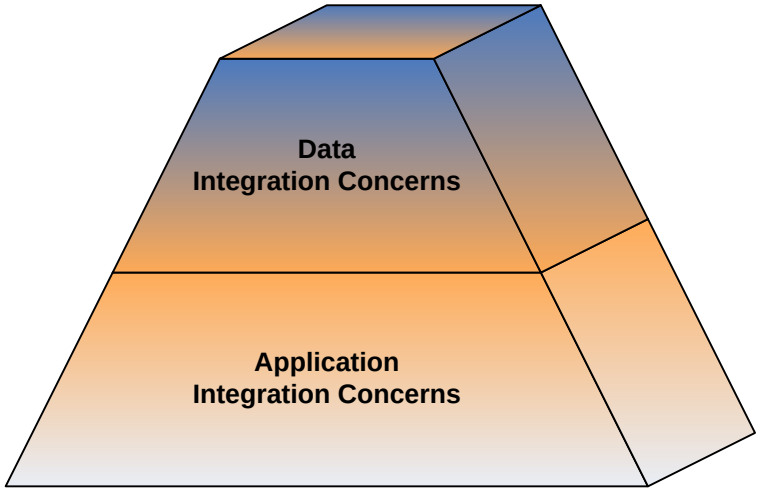
- Clear contract definition and separation of concerns
  - Application concerns



- **[Application]** performant, robust & guaranteed operation

Architecture Strategies

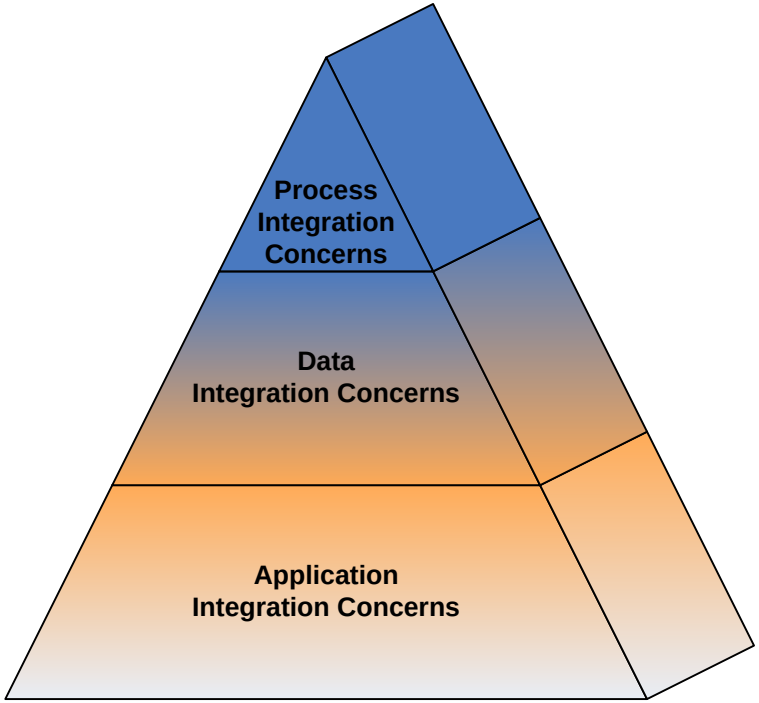
- Clear contract definition and separation of concerns
  - Application concerns
  - Data concerns



- **[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

Architecture Strategies

[Key Strategies]

▪ Single Point of Integration	▪ Loose Coupling	▪ Variety of Standard Contracts
▪ Separation of Concerns	▪ Tight Cohesion	▪ Standard Design Patterns
▪ Interchangeable Constructs	▪ Highly Testable Constructs	▪ Data Integrity
▪ Control & Management	▪ Monitor & Alerting	▪ Minimal Manual Intervention
▪ Support any Protocol/Entity	▪ Business Event Tracking	▪ Operational Audit

- Ensure traceability from architecture through to implementation
  - All too often architecture fails to translate into the real world
  - The right architecture can reduce complexity

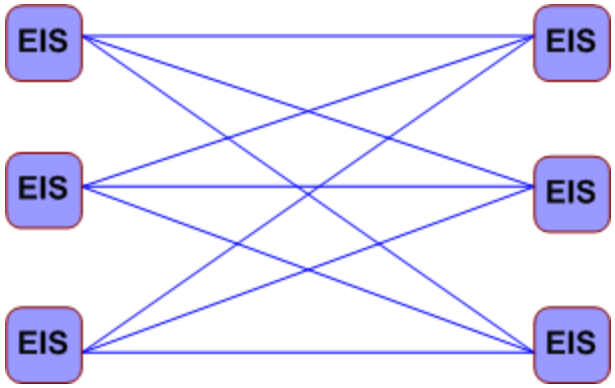
Translating the Architecture

[Key Strategies]

▪ Single Point of Integration	▪ Loose Coupling	▪ Variety of Standard Contracts
▪ Separation of Concerns	▪ Tight Cohesion	▪ Standard Design Patterns
▪ Interchangeable Constructs	▪ Highly Testable Constructs	▪ Data Integrity
▪ Control & Management	▪ Monitor & Alerting	▪ Minimal Manual Intervention
▪ Support any Protocol/Entity	▪ Business Event Tracking	▪ Operational Audit

[Issue] 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



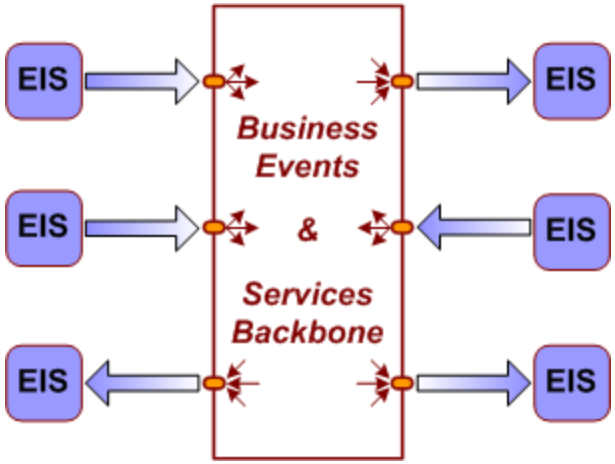
Translating the Architecture

[Key Strategies]

▪ Single Point of Integration	▪ Loose Coupling	▪ Variety of Standard Contracts
▪ Separation of Concerns	▪ Tight Cohesion	▪ Standard Design Patterns
▪ Interchangeable Constructs	▪ Highly Testable Constructs	▪ Data Integrity
▪ Control & Management	▪ Monitor & Alerting	▪ Minimal Manual Intervention
▪ Support any Protocol/Entity	▪ Business Event Tracking	▪ Operational Audit

[Concept] 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 localised *(assuming best practice)*



Translating the Architecture

[Key Strategies]

▪ Single Point of Integration	▪ Loose Coupling	▪ Variety of Standard Contracts
▪ Separation of Concerns	▪ Tight Cohesion	▪ Standard Design Patterns
▪ Interchangeable Constructs	▪ Highly Testable Constructs	▪ Data Integrity
▪ Control & Management	▪ Monitor & Alerting	▪ Minimal Manual Intervention
▪ Support any Protocol/Entity	▪ Business Event Tracking	▪ Operational Audit

[First Step] Modelling the Business Domain

- Defines fundamental entities, constructs, and their relationships
- Leverage existing domain knowledge
- External domain knowledge – we are not the first…
  - Enterprise Integration Design Patterns
    - [Ref. Enterprise Integration Patterns, Gregor Hohpe & Bobby Woolf, 2003]
  - Enterprise Application Design Patterns
    - [Ref. Patterns of Enterprise Application Architecture, Martin Fowler]

- Goal is a Rich Domain Model (data & behavior)
  - Anemic Domain Models (data only) fail over time

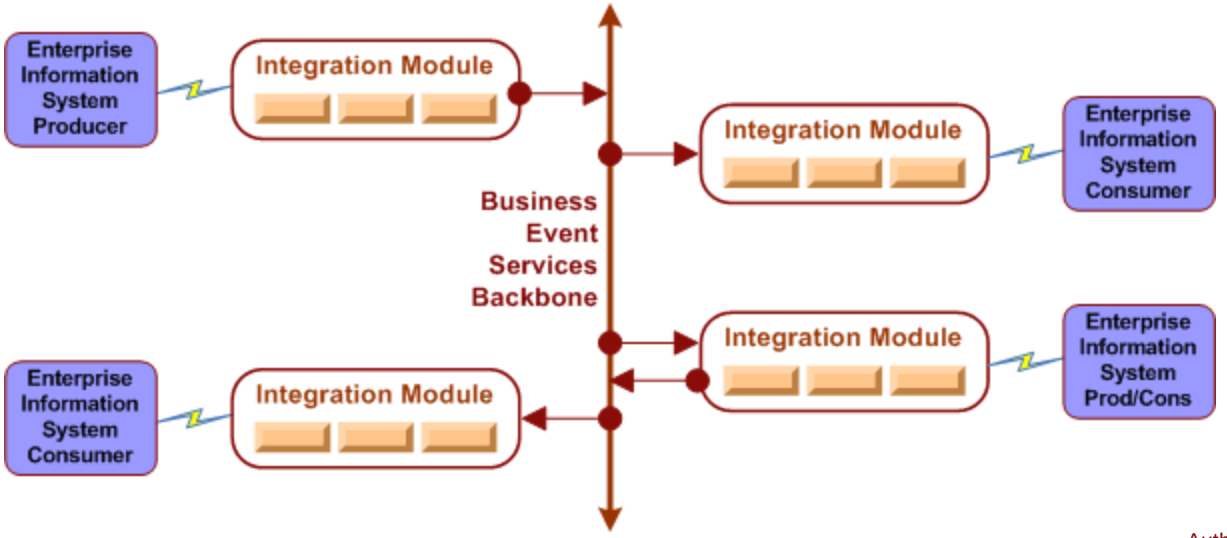
Translating the Architecture

[Key Strategies]

✓ Single Point of Integration	✓ Loose Coupling	✓ Variety of Standard Contracts
▪ Separation of Concerns	▪ Tight Cohesion	▪ Standard Design Patterns
▪ Interchangeable Constructs	▪ Highly Testable Constructs	▪ Data Integrity
▪ Control & Management	▪ Monitor & Alerting	▪ Minimal Manual Intervention
▪ Support any Protocol/Entity	▪ Business Event Tracking	▪ Operational Audit

[Construct] Integration Modules

- Provides a logical grouping of business operations as a single integration point
- Provide either a source, target, or bi-directional business flow.



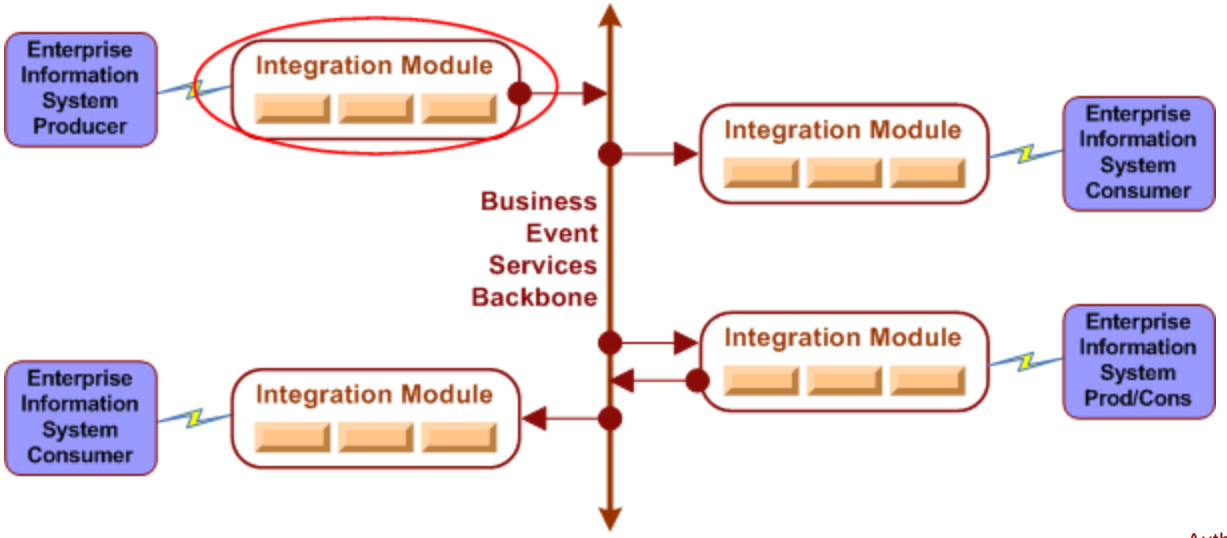
Translating the Architecture

[Key Strategies]

✓ Single Point of Integration	✓ Loose Coupling	✓ Variety of Standard Contracts
▪ Separation of Concerns	▪ Tight Cohesion	▪ Standard Design Patterns
▪ Interchangeable Constructs	▪ Highly Testable Constructs	▪ Data Integrity
▪ Control & Management	▪ Monitor & Alerting	▪ Minimal Manual Intervention
▪ Support any Protocol/Entity	▪ Business Event Tracking	▪ Operational Audit

[Construct] Integration Modules

- Provides a logical grouping of business operations as a single integration point
- Provide either a source, target, or bi-directional business flow.



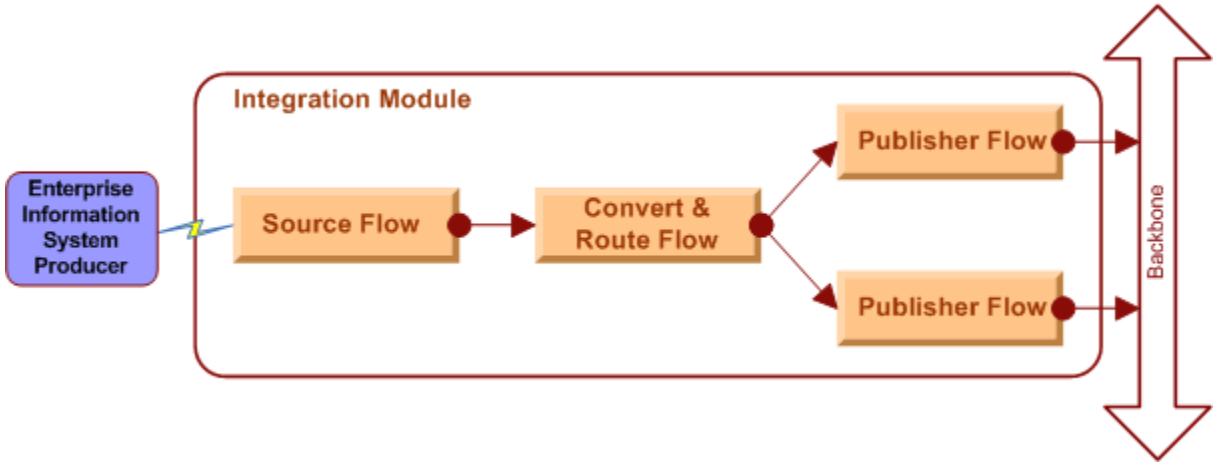
Translating the Architecture

[Key Strategies]

✓ Single Point of Integration	✓ Loose Coupling	✓ Variety of Standard Contracts
✓ Separation of Concerns	✓ Tight Cohesion	✓ Standard Design Patterns
▪ Interchangeable Constructs	▪ Highly Testable Constructs	▪ Data Integrity
▪ Control & Management	▪ Monitor & Alerting	▪ Minimal Manual Intervention
▪ Support any Protocol/Entity	▪ Business Event Tracking	▪ Operational Audit

[Construct] Flows

- Provide cohesive operations on a business artifact as a synchronous operation
- Multiple flows can be chained to isolate concerns
- Standard event container allows any data type to be transported



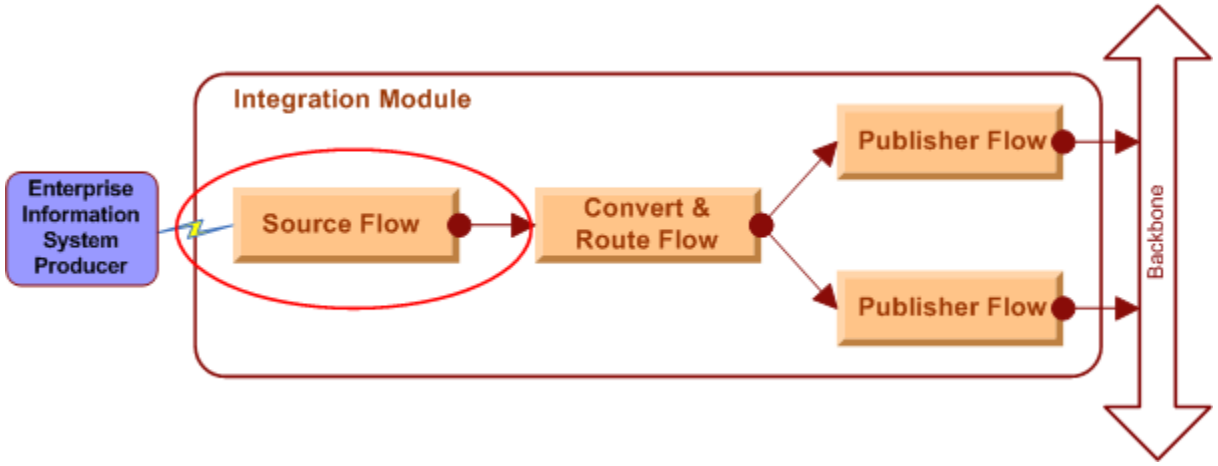
Translating the Architecture

[Key Strategies]

✓ Single Point of Integration	✓ Loose Coupling	✓ Variety of Standard Contracts
✓ Separation of Concerns	✓ Tight Cohesion	✓ Standard Design Patterns
▪ Interchangeable Constructs	▪ Highly Testable Constructs	▪ Data Integrity
▪ Control & Management	▪ Monitor & Alerting	▪ Minimal Manual Intervention
▪ Support any Protocol/Entity	▪ Business Event Tracking	▪ Operational Audit

[Construct] Flows

- Provide cohesive operations on a business artifact as a synchronous operation
- Multiple flows can be chained to isolate concerns
- Standard event container allows any data type to be transported



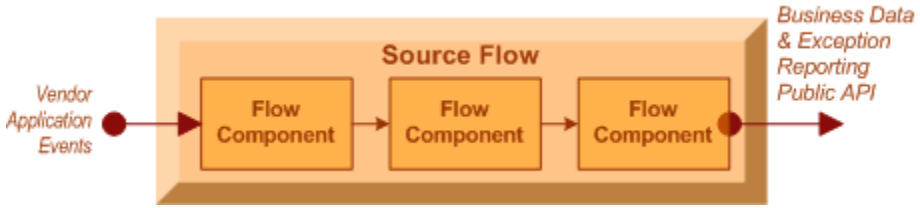
Translating the Architecture

[Key Strategies]

✓ Single Point of Integration	✓ Loose Coupling	✓ Variety of Standard Contracts
✓ Separation of Concerns	✓ Tight Cohesion	✓ Standard Design Patterns
✓ Interchangeable Constructs	✓ Highly Testable Constructs	✓ Data Integrity
▪ Control & Management	▪ Monitor & Alerting	▪ Minimal Manual Intervention
▪ Support any Protocol/Entity	▪ Business Event Tracking	▪ Operational Audit

[Construct] Flow Components

- POJO implementations provide independent, interchangeable operations
- Isolated component white box testing and chained black box testing
- All flow components within an atomic operation guaranteeing data integrity
- Unfettered event transport without casting or serialization overheads



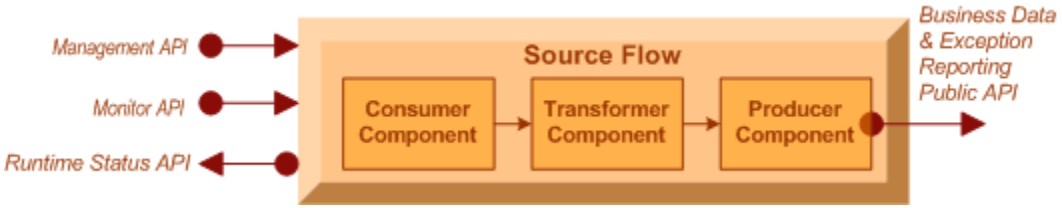
Translating the Architecture

[Key Strategies]

✓ Single Point of Integration	✓ Loose Coupling	✓ Variety of Standard Contracts
✓ Separation of Concerns	✓ Tight Cohesion	✓ Standard Design Patterns
✓ Interchangeable Constructs	✓ Highly Testable Constructs	✓ Data Integrity
✓ Control & Management	✓ Monitor & Alerting	✓ Minimal Manual Intervention
▪ Support any Protocol/Entity	▪ Business Event Tracking	▪ Operational Audit

[Construct] Flow Contracts

- Management API provides business flow control (start, stop, pause, resume)
  - Pause/Resume for fine grained consumer component resource management only
  - Stop/Start for course grained general component resource management
- Runtime Status API provides real-time health status (running, stopped, recovery, stoppedInError)
- Monitor API provides pluggable runtime health notifiers i.e. (Email, Zenoss, Hobbit, SMS, etc)
- Recovery Manager for configurable exception handling for automated flow recovery
  - Invocable by any component to change flow state



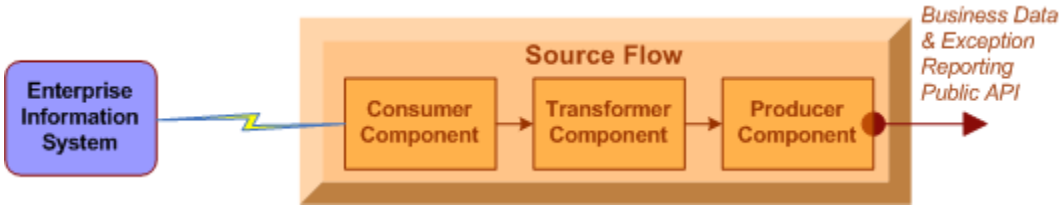
Translating the Architecture

[Key Strategies]

✓ Single Point of Integration	✓ Loose Coupling	✓ Variety of Standard Contracts
✓ Separation of Concerns	✓ Tight Cohesion	✓ Standard Design Patterns
✓ Interchangeable Constructs	✓ Highly Testable Constructs	✓ Data Integrity
✓ Control & Management	✓ Monitor & Alerting	✓ Minimal Manual Intervention
✓ Support any Protocol/Entity	▪ Business Event Tracking	▪ Operational Audit

[Construct] EIS / Application Integration

- Consumers can support any type of protocol / API
- Extend guaranteed integrity to non-guaranteed EIS sources/targets
- No proprietary knowledge required; only EIS API
- Resource adapters pluggable re-usable components (Java Connector Architecture spec)
- 3rd party off-the-shelf or internally developed



Translating the Architecture

[Key Strategies]

✓ Single Point of Integration	✓ Loose Coupling	✓ Variety of Standard Contracts
✓ Separation of Concerns	✓ Tight Cohesion	✓ Standard Design Patterns
✓ Interchangeable Constructs	✓ Highly Testable Constructs	✓ Data Integrity
✓ Control & Management	✓ Monitor & Alerting	✓ Minimal Manual Intervention
✓ Support any Protocol/Entity	✓ Business Event Tracking	✓ Operational Audit

[Construct] Administration Console

- Flow management
  - runtime status (stopped, running, recovery, stoppedInError)
  - runtime control (start, stop, pause, resume)
  - deployment control (start-up type {auto, manual, disabled})
  - flow component chain view and drill down
  - audit log of flow actions (stop/start/pause/resume)
- Wiretap management
  - dynamic creation/deletion of real-time in-flight data capture
- User management
  - creation, deletion, and permission maintenance

Translating the Architecture

[Key Strategies]

✓ Single Point of Integration	✓ Loose Coupling	✓ Variety of Standard Contracts
✓ Separation of Concerns	✓ Tight Cohesion	✓ Standard Design Patterns
✓ Interchangeable Constructs	✓ Highly Testable Constructs	✓ Data Integrity
✓ Control & Management	✓ Monitor & Alerting	✓ Minimal Manual Intervention
✓ Support any Protocol/Entity	✓ Business Event Tracking	✓ Operational Audit

[Construct] User Console

- In-flight business event searches
  - by module(s)
  - by flow(s)
  - by business stream(s)
  - parameter filters
    - business content, date/time, names, etc
- Event "*Life Identifier*" support for full end to end business event tracking
  - flow component chain view and drill down
  - audit log of flow actions (stop/start/pause/resume)
- User management
  - creation, deletion, and permission maintenance

Translating the Architecture

[Key Strategies]

✓ Single Point of Integration	✓ Loose Coupling	✓ Variety of Standard Contracts
✓ Separation of Concerns	✓ Tight Cohesion	✓ Standard Design Patterns
✓ Interchangeable Constructs	✓ Highly Testable Constructs	✓ Data Integrity
✓ Control & Management	✓ Monitor & Alerting	✓ Minimal Manual Intervention
✓ Support any Protocol/Entity	✓ Business Event Tracking	✓ Operational Audit

[Construct] Monitors & Notifiers

- Big Brother / Hobbit client
- Zenoss
- Email client
- Log scraper
- HTTP client

Translating the Architecture

[Key Strategies]

✓ Single Point of Integration	✓ Loose Coupling	✓ Variety of Standard Contracts
✓ Separation of Concerns	✓ Tight Cohesion	✓ Standard Design Patterns
✓ Interchangeable Constructs	✓ Highly Testable Constructs	✓ Data Integrity
✓ Control & Management	✓ Monitor & Alerting	✓ Minimal Manual Intervention
✓ Support any Protocol/Entity	✓ Business Event Tracking	✓ Operational Audit

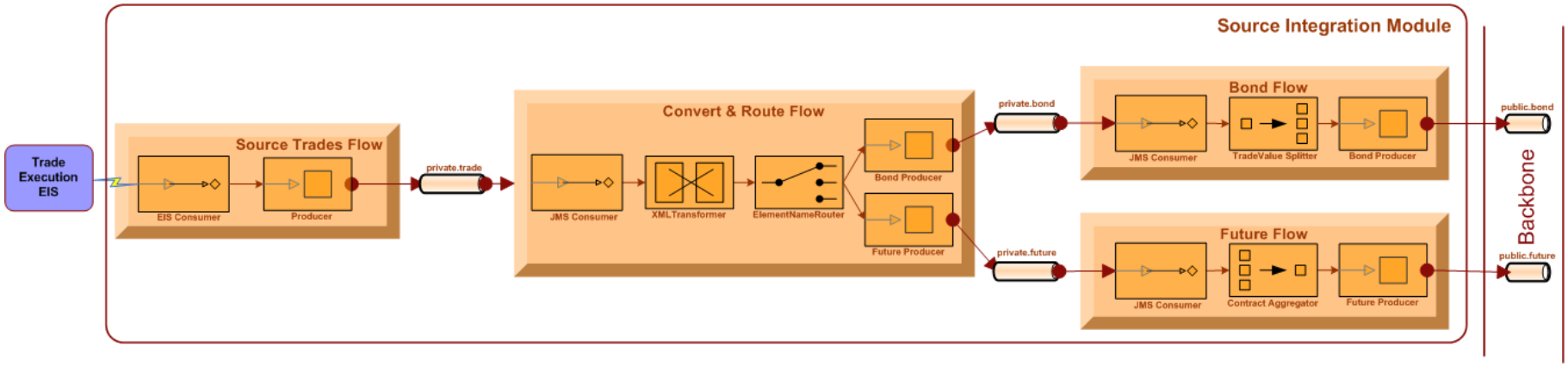
[Construct] OSS Test Frameworks & Ikasan FlowTestHarness

- Flow component automated unit testing
  - Junit, JMock, XMLUnit
- Initiator automated unit testing
  - JUnit, JMock
- Flow automated unit (integration) testing
  - Spring-test, Ikasan FlowTestHarness, HSQL & H2 in memory database
- Business Stream end to end testing
  - FitNesse

Translating the Architecture

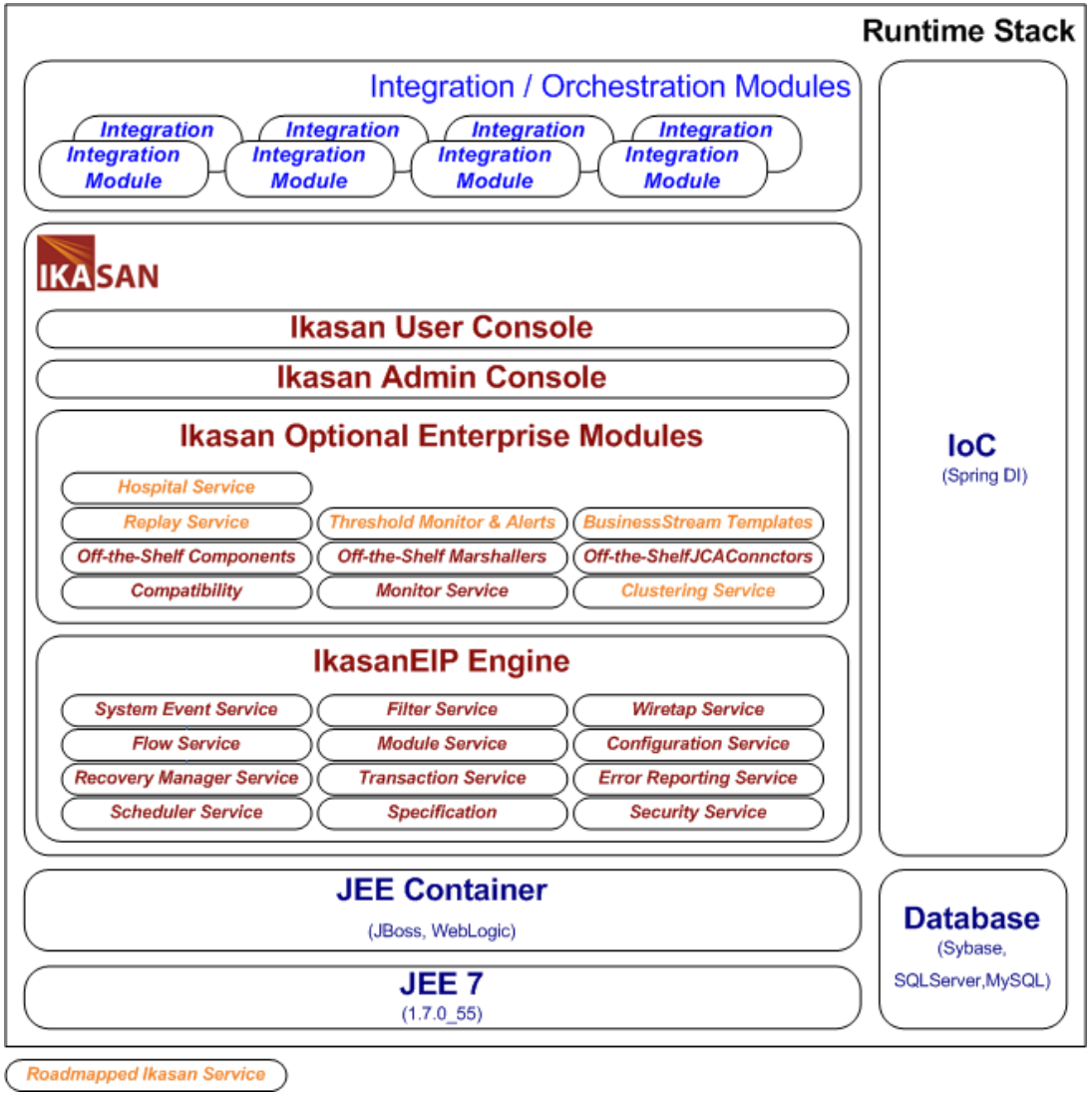
- Problem Domain
- Architecture Strategies
- **Translating Architecture**
- Technology Stack
- Features
- Development Lifecycle
- Roadmap
- Summary

- Sample Source Integration Module
  - Source Trades Flow
    - Get trades (CSV format) from EIS with minimal failure potential
  - Convert & Route Flow
    - Transform CSV -> XML syntax; Transform XML semantics; Route on type
  - Bond Flow
    - Split one bond event into many bond events based on trade size rule
  - Future Flow
    - Aggregate multiple future contract events until "*complete*" then push as single event



Technology Stack

- Runtime Stack



Technology Stack

- Development Stack
  - Core technologies
    - Java 1.7+ / JEE
    - XML
    - Spring 3.1.1.RELEASE
    - Hibernate
  - Test Frameworks
    - JUnit
    - Jmock
    - FitNesse
    - SpringTest
    - Ikasan Flow Test Harness
  - Scaffolding
    - Source Management – GitHub
    - Requirements and Issue Management – JIRA
    - Peer Review and Acceptance Management – Crucible/FishEye
    - Project Build Management – Maven
    - Project Artefact Management – Sonotype Nexus
    - Continuous Integration – Jenkins/Bamboo

Technology Stack

- IkasanEIP Stack
  - IkasanEIP Engine
    - ***IkasanEIP*** - engine underpinning all operations and services
  - Ikasan Optional Enterprise Modules
    - ***Compatibility*** – provides runtime inter-op between Ikasan versions
    - ***Monitor Service*** – default monitor & notifiers
    - ***Off-the-Shelf Flow Components*** i.e. JSON <-> XML, XSLT, MongoDB, JMS, Quartz
    - ***Clustering Service*** *(beta)* – makes Ikasan cluster aware
    - ***Threshold Monitor & Alerts*** *(beta)* – business throughput monitoring
    - ***Replay Service*** *(dev)*
      - event record & replay
      - Module revert to point in time state & replay (TBC)
    - ***Hospital Service*** *(dev)*
      - Management of failed events for review/resolution resubmission/discard
      - Management of failed events & dependents
    - ***Business Stream Templates*** *(planning)*
      - Boilerplate commoditisation of standard business orchestration (TBC)
  - Ikasan Admin Console & Ikasan User Console

Technology Stack

- IkasanEIP Stack
  - IkasanEIP is agnostic to the Operating System i.e. Linux, Windows, OSX
  - IkasanEIP runs In Container
    - Can run in any JEE compliant container
    - Proven in WebLogic 10+, JBoss 6+
  - IkasanEIP runs Standalone (outside container)
    - Can run standalone with reduced Enterprise Services
  - IkasanEIP Persistence
    - Proven with JDBC implementations ie. Sybase, SQLServer, MySQL, HSQLDB, H2
- Version 0.9.4 current production release
- Stable production backbone since 2007

Features

- Technical Integration
  - Off-the-Shelf Protocols
    - JMS (HornetQ, JBossA-MQ, ActiveMQ, WebLogic, IBM WebSphere MQ)
    - FTP (transactional & chunking support)
    - SFTP (transactional & chunking support)
    - RDBMS (via Hibernate ORM, Sybase, MS SQL, MySQL, Oracle)
    - SMTP (Email)
    - HTTP(s)
  - Data Constructs
    - Event – Ikasan internal entity container
    - EventIdentifier – Ikasan internal Event Life Identifier
    - Timestamp – Ikasan internal Event creation timestamp
    - Payload – actual entity in-flight – can be anything

Features

- Functionality
  - Flow Runtime Management – control at the runtime thread level
    - Allows a client to stop, start, pause, resume, enable, disable, query runtime status
  - Flow Drivers
    - Real-time Event Driven
    - Batch Event Driven
    - Schedule Driven
  - Flow Non-Functional Features
    - Full Transactional Semantics
      - Non-transactional
      - Local transactions
      - 2-Phase transactions
      - Last resource commit optimisation (LRCO)
        - incorporate non-transactional resources within a distributed transaction
    - Configurable Automated Recovery
      - Rollback operations & stop
      - Rollback operations and retry in x seconds
      - Rollback operations and retry a number of times or indefinitely
      - Report the exception and continue

Features

- Functionality
  - Flow Components
    - Transformers
      - XSLT
      - XML Reader, Validator
      - any POJO implementing Translator/Converter interface
    - Routers
      - Recipient list
      - Content
      - Xpath
      - Any POJO implementing Ikasan Router interface
    - Sequencers
      - Aggregator
      - Any POJO implementing Ikasan Sequencer interface
    - Splitters
      - Any POJO implementing Ikasan Splitter interface
    - Filters
      - Any POJO implementing Ikasan FilterRule/Filter interface

Features

- Functionality
  - Flow Components
    - Endpoints
      - Consumers
        - JMS Standalone
        - JMS Container Based
        - Scheduled
        - Any resource adapter client or third-party API
        - Any POJO implementing Ikasan Consumer interface
      - Brokers
        - MongoDB
        - JDBC
        - Any resource adapter client or third-party API
        - Any POJO implementing Ikasan Broker interface
      - Producers
        - JMS Standalone
        - JMS Container Based
        - Loggers
        - Any resource adapter client or third-party API
        - Any POJO implementing Ikasan Producer interface
  - Dynamically capture runtime data at any point in the flow
  - Users able to search business content over Flows/Business Streams

Features

- Business Integration
  - Post Trade Execution
    - Bloomberg
      - Trade Feed
      - Consolidated Message Feed
      - Derivative Settlement Feed
      - Daily Trade Reporting
    - PATS Post Trade Feed
    - Espeed Post Trade feed
    - Tradeweb Post Trade Feed (US Treasuries & JGBs)
    - ION Post Trade Feed
    - Fidessa ETP Execution Report and Allocation FIX messages
    - ANVIL Repo Trading
  - Positions
    - ION
    - Bloomberg Position Feed
    - Intellimatch
  - Price Marks
    - ION (Price Marks)
    - Bloomberg (Price UMTM)

Features

- Business 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 Holidays
    - Fidessa ETP MIS & FDA
    - GoldenSource
  - Market Data
    - Markit CDS Service
    - Markit iBoxx Benchmark Indices
    - Bank of America Merrill Lynch Indices
    - Xenomorph Timescape

Features

- Business Integration
  - Settlements & Operations
    - SWIFT Alliance
    - GlobeOp
  - Ratings
    - Moodys
    - Standard & Poors
  - Compliance
    - Thomson Reuters TransWatch Securities Data
  - Corporate Actions
    - FTI Corporate Action & Securities Universe

Development Lifecycle

- Approach based on TDD/BDD Agile dogma
  - Gather/refine user stories and create use cases
  - Sketch the stories into a stream/module/flow design
  - Create/refine the Flow Test Harness reflecting the design
  - Create modules/flows/components to meet test harness acceptance
  - Peer/pair walk through (10 minute daily stroll)
    - Design reviews
    - Code reviews
  - Continuous Integration (CI) test coverage increases
  - Deliver functionality to users for proofs
  - Repeat!
- Recent Example Implementation Times
  - 1 day – Completed module delivery for raw to raw message transport (RMS: Raw Message Service);
  - 1 - 15 days – Completed module delivery for transformed XML (CMS: Common Message Service); real-time feed and batch aggregation;
  - 50 days – Completed delivery of complex business work flow (BPAWS: Business Process Application Workflow Service). Orchestrating Fixed Income Bond Trading flows between two business entities.

Roadmap

- Version 0.9.0 released on 29-July-2013
  - Main Features
    - Simplicity (0.9.0)
      - No more initiators
      - Clearer more modular structure to the Ikasan platform
    - Significant Performance Improvement (0.9.0)
      - Changed the nature of Event and payload to utilise Java Generics
      - Improved handling of exceptions
      - Improved recovery cycle management
    - Simplified Domain Model (0.9.0+)
      - Support any user defined entity in the domain model
      - Support of any user defined domain properties
- Version 0.9.4 (Latest and Stable)
- Version 1.0.0 due shortly, currently in beta

Roadmap

- Version 1.1.0 due Q3 2014
  - Threshold Monitor & Alerts
  - Clustering Service
  - Hospital Service
  - Replay Service
  - Greater focus on Developer Documentation
- Version 1.2.0 (TBC)
  - Dynamic Provisioning
  - Business Stream Templates

Summary

- Ikasan Project Resource Summary
  - Web Site: http://www.ikasan.org
  - GitHub: https://github.com/ikasanEIP/ikasan
  - Wiki: http://sourceforge.net/apps/mediawiki/ikasaneip/index.php?title=Main_Page
  - JIRA/Confluence/Bamboo: http://ikasan.atlassian.net
  - Mailing List: http://lists.sourceforge.net/lists/listinfo/ikasaneip-user
  - IRC: http://sourceforge.net/apps/mediawiki/ikasaneip/index.php?title=IRC

Open Source Enterprise Integration Platform

Jeff Mitchell & Jun Suetake
ikasan.org
July 2014