MIMOSA

Open Standards for
Physical Asset Management

OIIE Implementation Introduction

Publish/Subscribe over the ISBM

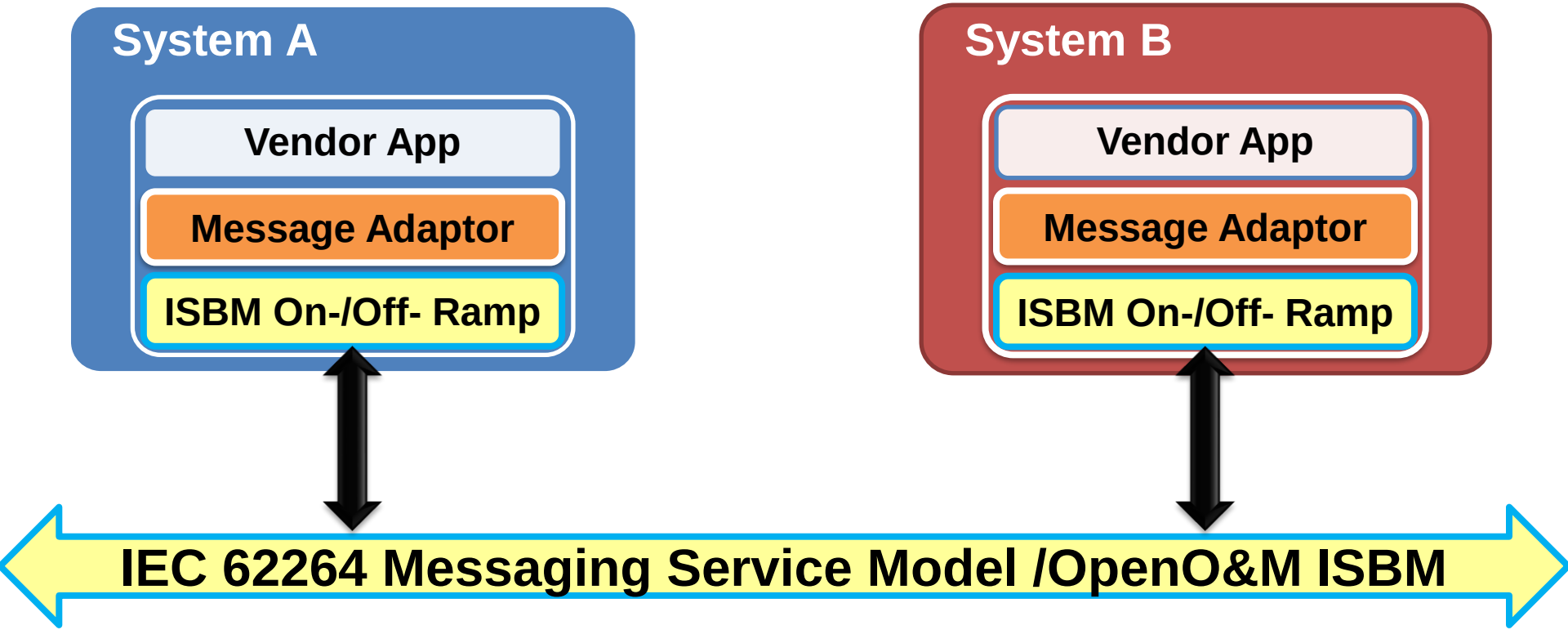
Matt Selway, Karamjit Kaur

University of South Australia

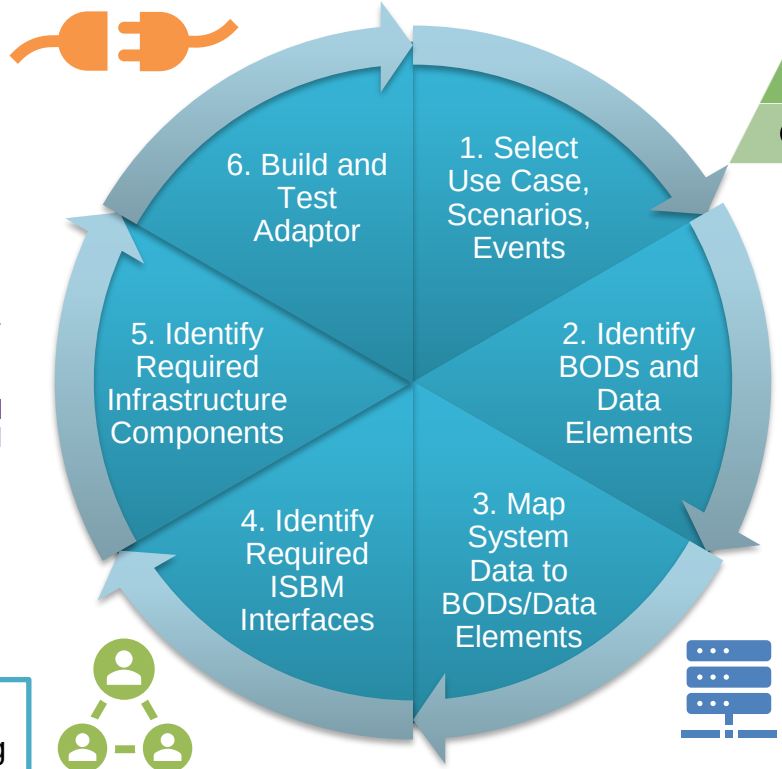
Introduction

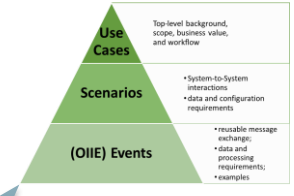
- Brief introduction to and demonstration of ISBM Pub/Sub interface
  - Future workshops add complexity, build to implementation of Use Case
- Where to find the ISBM demo instance for the OIIE AuWG?
  - Occasionally updates will be rolled out and may reset the DB
  - OpenAPI 3.0 Specification for REST interface
- Where to find supporting adaptor libraries?
  - **Ruby** Python Java C#/.Net
  - **REST**/SOAP TBD TBD V1/V2-TBD
- Links to resources at the end of the presentation

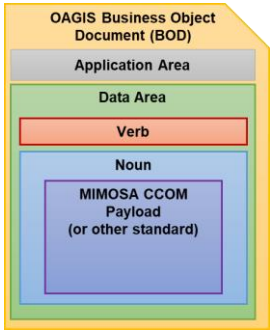
OIIE Adaptors



OIIE/ISBM Adaptor Development Process







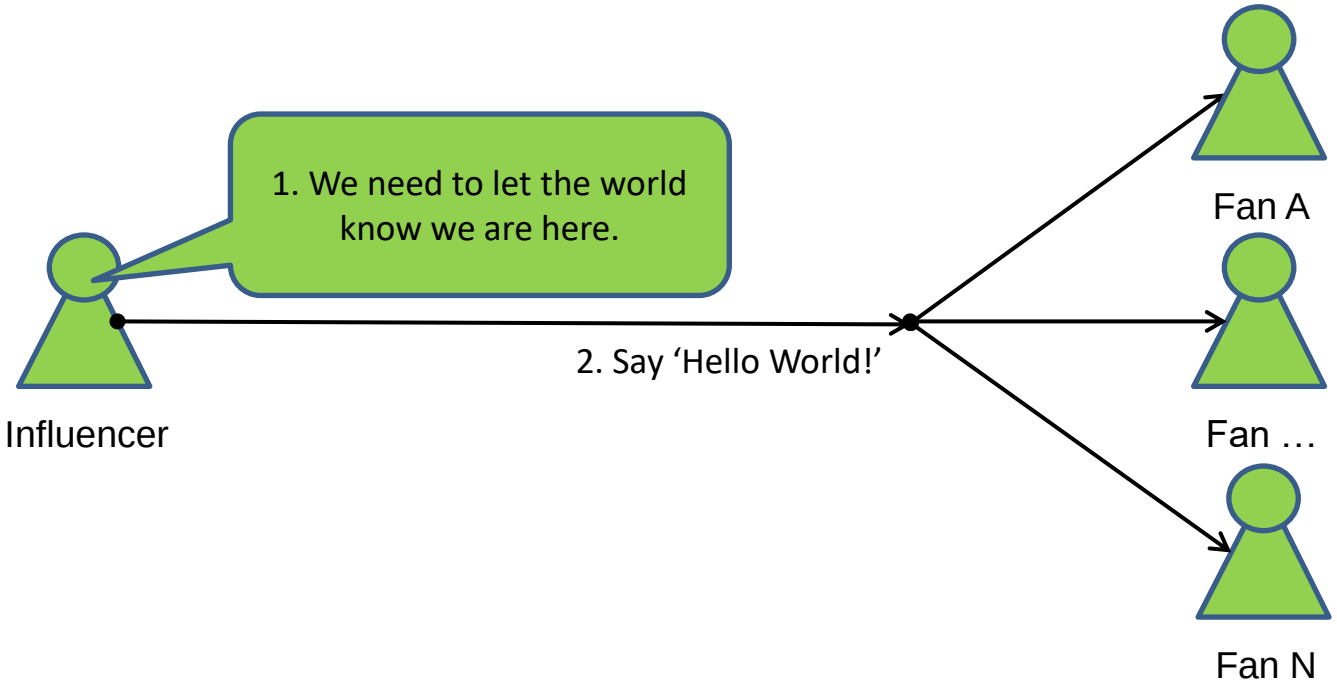
Refer to [Guide to developing an OIIE/ISBM Adapter] @ mimosa.org

Additional Requirements and Design Decisions

- What **role** does my application play?
- Which **transaction modes** (e.g., Pub/Sub, Request/Response) need to be supported?
- Which **interface type** is most appropriate (i.e., REST or SOAP)?
- Which **service methods** need to be supported?

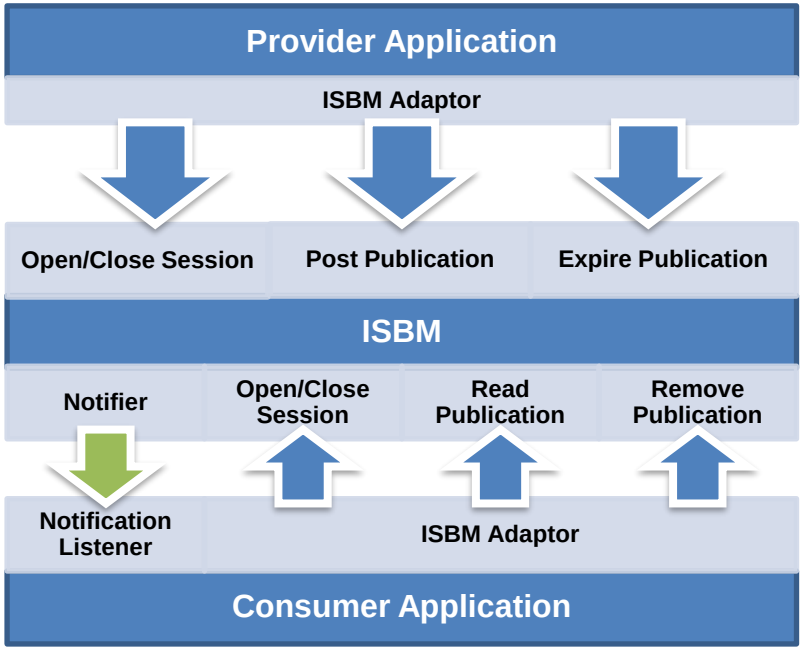
- Will my application support **asynchronous callback** or revert to polling?
- What events will **trigger a payload** to be sent on the ISBM?
- What events will be triggered when a **message is received**?
- How will the **ISBM connection be configured** (incl. channel, topic and token)?
- How are **other configuration items**, such as polling or retry intervals, configured?
- How are channels and topics **persisted** across application restarts?
- Where will ISBM activity be **logged and how are errors** presented to users?
- What ISBM activity needs to be persisted for **audit** purposes?

User Story—Hello World



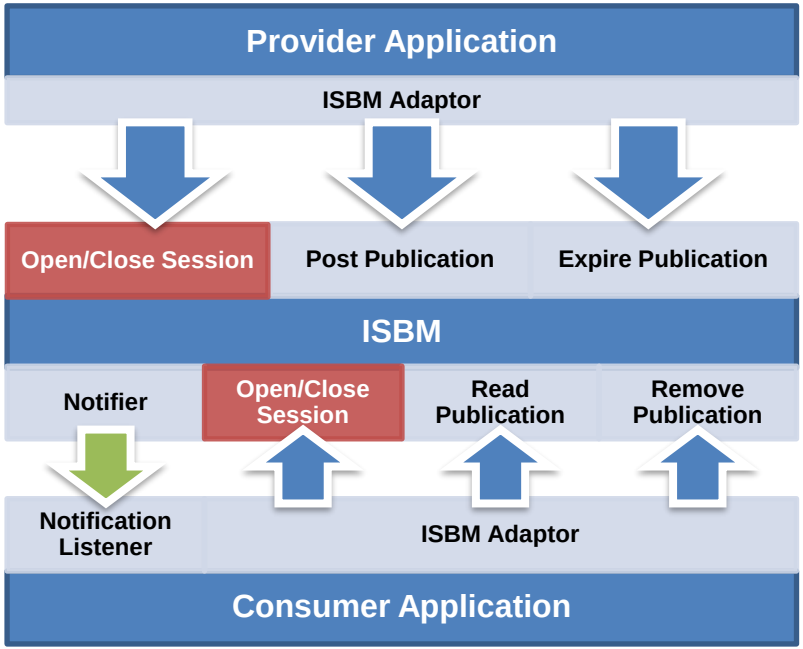
Publish/Subscribe Interface

Publish/Subscribe
(Notification optional)



Publish/Subscribe Interface

Publish/Subscribe
(Notification optional)

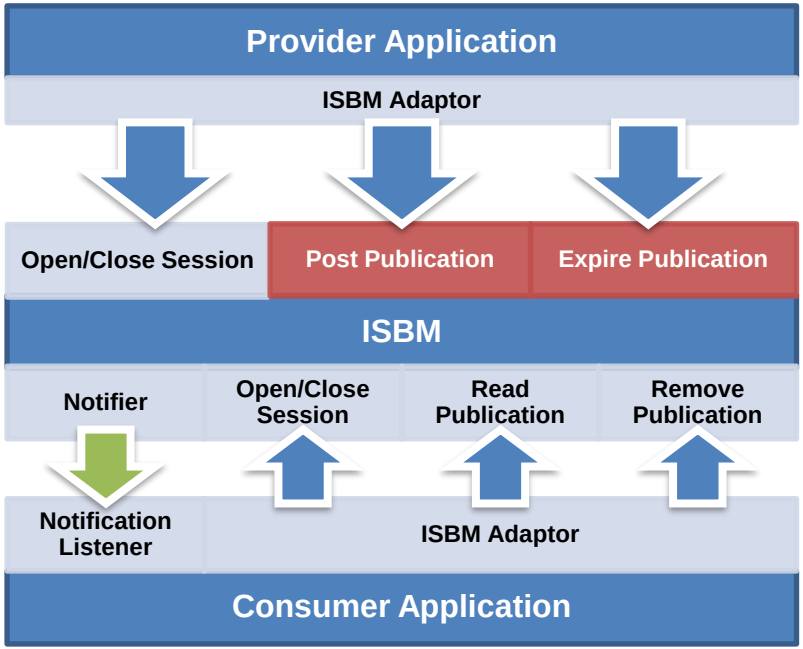


- Open Publication Session
  - ChannelURI
  - Returns: SessionID
- Open Subscription Session
  - ChannelURI
  - Topics
  - ListenerURL (optional)
  - FilterExpression (optional)
- Close Session
  - SessionID
  - Note: Closing Publication Session will expire all messages

Full details in the ISBM 2.0 Specification @ openoandm.org

Publish/Subscribe Interface

Publish/Subscribe
(Notification optional)

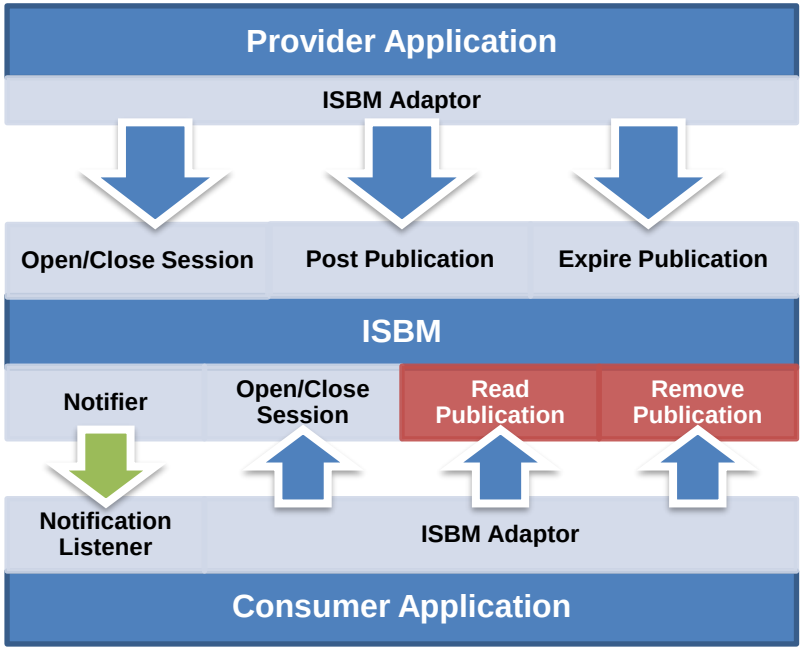


- Post Publication
  - SessionID
  - MessageContent
  - Topics
  - Expiry (optional)
  - Returns: MessageID
- Expire Publication
  - SessionID
  - MessageID

Full details in the ISBM 2.0 Specification @ openoandm.org

Publish/Subscribe Interface

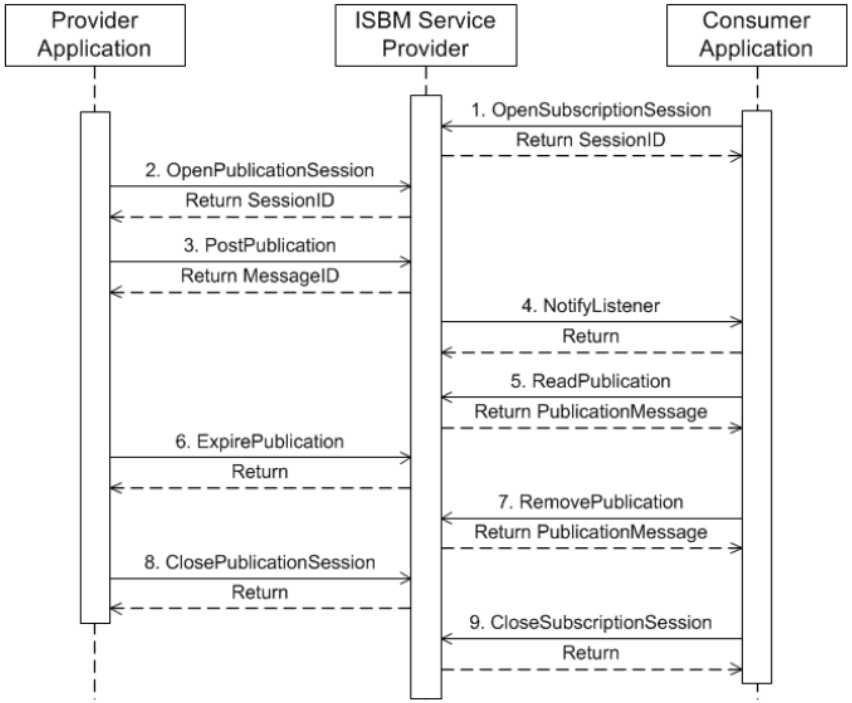
Publish/Subscribe
(Notification optional)



- Read Publication
  - SessionID
  - Returns: Message
    - MessageID
    - MessageContent
    - Topics
- Remove Publication
  - SessionID

Full details in the ISBM 2.0 Specification @ openoandm.org

Basic Flow



ISBM Instance/Demonstration

- First things first
  - OIIE AuWG ISBM Instance
  - “Hello World” Demo App (https://hello-world-isbm-demo.herokuapp.com/)
  - Interactive Web Interface (http://simpleisbm.demo.assetricity.com/demo/)
- Python Example (basic REST, no adaptor library)
- Ruby Example (using adaptor library)

Links and Resources

- ISBM 2.0 Specification – https://www.openoandm.org/isbm/
- Guide to Developing an OIIE/ISBM Adaptor – https://www.mimosa.org/guide-to-developing-an-isbm-adapter/
- OIIE AuWG ISBM instance – https://isbm.au-wg.oiiecosystem.net/
- REST Interface Interactive – http://simpleisbm.demo.assetricity.com/demo
  - change the source address to use the AuWG instance: 'https://isbm.au-wg.oiiecosystem.net/rest/api'
- "Hello World" Demo App – https://hello-world-isbm-demo.herokuapp.com/
- ISBM 2.0 Ruby Adaptor REST library – https://github.com/assetricity/isbm2_adaptor_rest