GeoEvent Server: 
Creating Connectors and Processors Using the GeoEvent SDK

Javier Delgadillo  
Real-Time GIS Developer, Esri  
jdelgadillo@esri.com

Ming Zhao  
Real-Time GIS Developer, Esri  
mzhao@esri.com
Agenda

1. Real Time GIS Overview
2. Connectors
3. Transports
4. Adapters
5. Processors
6. Summary
Real Time GIS Overview
ArcGIS as an IoT Platform

ArcGIS Enterprise with real-time & big data capabilities

- Ingest high velocity real-time IoT data into ArcGIS.
- Perform continuous analytics on IoT events as they are received.
- Store IoT observations in a spatiotemporal big data store.
- Visualize high velocity & volume IoT data:
  - as an aggregation
  - or as discrete features.
- Notify about IoT patterns of interest.
- Adjust behavior of things in our environment through actuation.
Ingestion of real-time data

GeoEvent Server: input connectors

GeoEvent Services

you can create your own connectors

Out of the Box

Poll an ArcGIS Server for Features
Poll an external website for GeoJSON, JSON, or XML
Receive Features, GeoJSON, JSON, or XML on a REST endpoint
Receive GeoJSON or JSON on a WebSocket
Receive RSS
Receive Text from a TCP or UDP Socket
Subscribe to an external WebSocket for GeoJSON or JSON
Watch a Folder for new CSV, GeoJSON, or JSON Files

Esri Gallery

ActiveMQ
AIS
Common Alerting Protocol
CoT
Cursor-on-Target
esd
Exploitation Support Data
GTFS
Instagram
Kafka
KML
MQTT
NMEA 0183
RabbitMQ
Sierra Wireless (RAP)
Trimble (TAIP)
Twitter

Partner Gallery

AWS IoT
Amazon IoT
Azure IoT
CompassCom
CompassLDE
enviroCar
exactEarth AIS
FAA (ASDI)
GNIP
Networkfleet
OSIsoft
Valarm
Waze
Zonar
Zonar
you can create your own connectors

Ingestion of real-time data
configure a new input connector by pairing a transport & adapter together

**GeoEvent Server**

**GeoEvent Services**

**Inputs**

- GeoJSON
- JSON

**Outputs**

- GeoMessage
- Instagram
- Twitter

**Connectors**

<table>
<thead>
<tr>
<th>Transport</th>
<th>Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kafka</td>
<td>Feature-JSON</td>
</tr>
<tr>
<td>HTTP</td>
<td>GeoJSON</td>
</tr>
</tbody>
</table>

**Transports**

- Feature Service
- File
- HTTP
- HTTP+BasicAuth
- HTTP+OAuth
- TCP
- UDP
- Waze
- WebSocket

**Adapters**

- Feature-JSON
- GeoJSON
- JSON
- RSS
- Text
- XML

**Gallery**

- Amazon IoT
- Azure IoT
- Kafka
- IRC
- Kafka
- MQTT
- RabbitMQ
- Twitter

**Input Connector = Adapter + Transport**

Ingestion of real-time data
Dissemination
GeoEvent Server: output connectors

GeoEvent Server
GeoEvent Services
Inputs
Outputs

you can create your own connectors

Out of the Box
Add or Update a Feature
Publish Text to a UDP Socket
Push GeoJSON or JSON to an external Website
Push GeoJSON or JSON to an external WebSocket
Push Text to an external TCP Socket
Send a Text Message
Send an Email
Send an Instant Message
Send Features to a Stream Service
Write to a CSV, GeoJSON, or JSON File
Add a Feature to a spatiotemporal big data store
Update a feature in a spatiotemporal big data store

Esri Gallery
AWS IoT
Amazon IoT
Azure IoT
ActiveMQ
CoT
Cursor-on-Target
Hadoop
Kafka
MongoDB
MQTT
RabbitMQ
Twitter
Dissemination

configure a new output connector by pairing an adapter & transport together

you can create your own connectors

GeoEvent Server

Inputs

GeoEvent Services

Outputs

Connectors

<table>
<thead>
<tr>
<th>Connectors</th>
<th>Adapter</th>
<th>Transport</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send an Email</td>
<td>Message Formatter</td>
<td>SMTP</td>
</tr>
<tr>
<td>Publish Feature-JSON to Kafka</td>
<td>Feature-JSON</td>
<td>Kafka</td>
</tr>
</tbody>
</table>

Adapters

<table>
<thead>
<tr>
<th>Adapters</th>
<th>Cache</th>
<th>Feature-JSON</th>
<th>GeoJSON</th>
<th>JSON</th>
<th>Message Formatter</th>
<th>RSS</th>
<th>Text</th>
<th>WebSocket</th>
</tr>
</thead>
</table>

Transports

<table>
<thead>
<tr>
<th>Transports</th>
<th>Big Data Store</th>
<th>Feature Service</th>
<th>Stream Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMTP</td>
<td>SMTP</td>
<td>SMTP</td>
<td>SMTP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Transports</th>
<th>File</th>
<th>HTTP</th>
<th>HTTP+BasicAuth</th>
<th>HTTP+OAuth</th>
<th>SMS</th>
<th>Cache</th>
<th>Feature-JSON</th>
<th>GeoJSON</th>
<th>JSON</th>
<th>Message Formatter</th>
<th>RSS</th>
<th>Text</th>
<th>WebSocket</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>TCP</td>
<td>TCP</td>
<td>TCP</td>
<td>TCP</td>
<td>TCP</td>
<td>TCP</td>
<td>TCP</td>
<td>TCP</td>
<td>TCP</td>
<td>TCP</td>
<td>TCP</td>
<td>TCP</td>
<td>TCP</td>
</tr>
<tr>
<td>UDP</td>
<td>UDP</td>
<td>UDP</td>
<td>UDP</td>
<td>UDP</td>
<td>UDP</td>
<td>UDP</td>
<td>UDP</td>
<td>UDP</td>
<td>UDP</td>
<td>UDP</td>
<td>UDP</td>
<td>UDP</td>
<td>UDP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Transports</th>
<th>XSS</th>
<th>Cache</th>
<th>Feature-JSON</th>
<th>GeoJSON</th>
<th>JSON</th>
<th>Message Formatter</th>
<th>RSS</th>
<th>Text</th>
<th>WebSocket</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSL</td>
<td>SSL</td>
<td>SSL</td>
<td>SSL</td>
<td>SSL</td>
<td>SSL</td>
<td>SSL</td>
<td>SSL</td>
<td>SSL</td>
<td>SSL</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Transports</th>
<th>ActiveMQ</th>
<th>Amazon IoT</th>
<th>Azure IoT</th>
<th>Hadoop</th>
<th>IRC</th>
<th>Kafka</th>
<th>MongoDB</th>
<th>MQTT</th>
<th>RabbitMQ</th>
<th>TCP-Squirt</th>
<th>Twitter</th>
</tr>
</thead>
</table>

Esri Gallery

Cursor-on-Target
Streaming analytics, policies & orchestration

- GeoEvent Server: processors

You can create your own processors

GeoEvent Server

GeoEvent Services

Inputs

Outputs

GeoEvent Services:
- Buffer Creator
- Convex Hull Creator
- Difference Creator
- Envelope Creator
- Field Calculator
- Field Enricher
- Field Mapper
- Field Reducer
- Geotagger
- Incident Detector
- Intersector
- Projector
- Simplifier
- Symmetric Difference
- Track Gap Detector
- Union Creator

Out of the Box

Esri Gallery

- Add XYZ
- Bearing
- Ellipse
- Event Volume Control
- Extent Enricher
- Field Grouper
- GeoNames Lookup
- Motion Calculator
- Range Fan
- Reverse Geocoder
- Service Area Creator
- Symbol Lookup
- Track Idle Detector
- Unit Converter
- Visibility
- Query Report
Extending GeoEvent
Software Development Kit (SDK)

• You can create your own custom connectors and processors using the GeoEvent Software Development Kit (SDK)
GeoEvent Server SDK

- **api:** JavaDoc content associated with GeoEvent Server SDK
- **repository:** Local maven repository
- **samples:** Sample connectors (and processors)
- **GeoEvent Developer Guide**
Extending GeoEvent
Software Development Kit (SDK)

- JavaDoc available by opening api/index.html in a browser (Chrome, Firefox, IE, Safari)
2 Connectors
Connectors
What Does a Connector Do?

- **Connectors** are used to create **Inputs** and **Outputs**, hiding the technical details.
- It might be very specific
  - Get latest earthquakes from USGS
- Or more general
  - Connect to an RSS feed
Creating Inputs
Configured Using Connectors

- National Weather Service
- US Geological Survey
- Live Traffic Services

Connectors:
- Receive Data

GeoEvent Server:
- GeoEvent Services
  - Storms
  - Earthquakes
  - Traffic

Outputs:
- Receive RSS
- Poll External Website for JSON
- Receive Text from TCP
Connectors
Configuring a Connector

- The connector helps the user by
  - Providing default values
  - Re-label properties to be appropriate to the context
  - Move properties under an “advanced” area to discourage modification
  - Completely hide properties that the user should not see
By choosing a **Connector**, the user implicitly selects **components** from the GeoEvent Server that knows:

- HOW to move data (**Transport**)
- WHAT the data looks like (**Adapter**)

**Example Input**

**Input** *(Receive text from a TCP Socket)*

- **Start**
- **Stop**
- **Byte [ ]**

**TCP Transport**  **Text Adapter**  **GeoEvent**  **GeoEvent Services**
Demo
Exploring a Connector
3 Transports
Transport
What makes up a Transport

- Transport Service
  - Provides a Transport Definition
  - Instantiates new Transports

- Transport Definition
  - Defines transport metadata and properties

- Transport
  - Implements transport application logic
Transport Behavior

Transport

- Transports are given
  - Properties that define behavior
  - A “ByteListener” where bytes should be sent
- Transport is started by the server and it sends bytes to the receiver
- Transport is stopped by the server and it stops sending bytes
Transports have a lifecycle that determines if they are producing data.

- **STOPPED**: Server calls `start()`.
- **STARTING**: Transport reads properties and starts.
- **STARTED**: Internal error occurs.
- **STOPPING**: Server calls `stop()`.
- **ERROR**: Transport closes resources.
Outbound Transport

• Outbound Transports accept arrays of bytes from the Adapter and transmit them.
• Occasionally the destination for the bytes depends on content in the GeoEvent.
  - The Transport has the option of “looking back” at the GeoEvent that generated the bytes, and using it to route the data.
Property Definition

Properties

• Transports and Adapters request properties through their “Definition” class

• Each requested property has a
  - Name, Description, Type (String, Integer, Float, …), Default Value …
Advanced Topics – More on Properties

Properties

• Properties can be Mandatory or Optional.
• Some properties are dependent on others
  ▪ Example: “Compression Algorithm” depends on “Compressed = True”
• Some properties only accept values from a list of Allowed Values
  ▪ “Compression Algorithm” : [ Run Length Encoding, Zip, LZW ]
Properties can also be created programmatically in a Java Class.

Allowed values were strings in previous release. Replaced by objects in 10.3.1. As a result, you can display allowed values with localized labels.

```java
List<LabeledValue> compressionAlgorithmAllowedValues = new ArrayList<LabeledValue>(6);
compressionAlgorithmAllowedValues.add(new LabeledValue("${sample.gep.sample-adapter-properties.COMPRESSION_RLE_LBL}","RLE");
compressionAlgorithmAllowedValues.add(new LabeledValue("${sample.gep.sample-adapter-properties.COMPRESSION_ZIP_LBL}","ZIP");
compressionAlgorithmAllowedValues.add(new LabeledValue("${sample.gep.sample-adapter-properties.COMPRESSION_LZW_LBL}","LZW");

PropertyDefinition compressionAlgorithms = new PropertyDefinition( COMPRESSION_ALGORITHM_PROPERTY, PropertyType.String, "ZIP",
"${sample.gep.sample-adapter-properties.ADPATER_IN_FAV_COLOR_LBL}",
"${sample.gep.sample-adapter-properties.ADPATER_IN_FAV_COLOR_DESC}",
"compressOutput=true", false, false, compressionAlgorithmAllowedValues );
propertyDefinitions.put( COMPRESSION_ALGORITHM_PROPERTY, compressionAlgorithms );
```
Demo
Exploring a Custom Transport
Adapter

What makes up an Adapter

- Adapter Service
  - Provides an Adapter Definition
  - Instantiates new Adapters

- Adapter Definition
  - Defines adapter metadata and properties

- Adapter
  - Implements adapter application logic
Adapter Behavior
Adapters

- Adapters are given
  - Properties that define behavior
  - A “GeoEventListener” where the GeoEvents should be sent
- Adapters are DATA DRIVEN
  - No start/stop calls
  - The adapter is handed a byte array and pushes any generated GeoEvents to the Listener
public class SampleInboundAdapterDefinition extends AdapterDefinitionBase
{
    public SampleInboundAdapterDefinition()
    {
        super(AdapterType.INBOUND);
        try
        {
            GeoEventDefinition md = new DefaultGeoEventDefinition();
            md.setName("SampleGeoEventDefinition");
            md.setAccessType(AdapterType.editable);
            List<FieldDefinition> fieldDefinitions = new ArrayList<FieldDefinition>();
            fieldDefinitions.add(new DefaultFieldDefinition("track_id", FieldType.Integer));
            fieldDefinitions.add(new DefaultFieldDefinition("location", FieldType.Geometry, "GEOMETRY"));
            md.setFieldDefinitions(fieldDefinitions);
            geoEventDefinitions.put(md.getName(), md);
        }
        catch (ConfigurationException ex)
        {
        }
    }
Adapter Code Walkthrough

Adapter Properties

```java
@override
public String getName()
{
    return "Sample";
}
@override
public String getLabel()
{
    return "${sample.gep.sample-adapter.ADAPTER_IN_LABEL}";
}
@override
public String getDomain()
{
    return "sample.adapter.inbound";
}
```
Adapter Code Walkthrough

Parsing Data

```java
@override
public GeoEvent adapt(ByteBuffer buffer, String channelId) {
    buffer.mark();
    try {
        int id = buffer.getInt();
        double x = buffer.getDouble();
        double y = buffer.getDouble();
        GeoEvent msg;
        try {
            msg = geoEventCreator.create(((AdapterDefinition) definition).getGeoEventDefinition("SampleGeoEventDefinition"), guid);
            LOGGER.info("CREATED MSG");
        } catch (MessagingException e) {
            return null;
        }
        msg.setField("track_id", id);
        msg.setGeometry(new MapGeometry(new Point(x, y), SpatialReference.create(4326)));
        return msg;
    } catch (BufferUnderflowException ex) {
        buffer.reset();
        return null;
    } catch (FieldException e) {
        LOGGER.error("FIELD_SET_ERROR", e, e.getMessage());
    }
    return null;
}
```
5 Processors
Real-Time GIS

ArcGIS GeoEvent Extension for Server: continuous analytics on GeoEvents as they are received

GeoEvent Extension

Inputs

GeoEvent Services

Outputs

You can create your own processors.

Out of the Box

Buffer Creator
Convex Hull Creator
Difference Creator
Envelope Creator
Field Calculator
Field Enricher
Field Mapper
Field Reducer

Geotagger
Incident Detector
Intersector
Projector
Simplifier
Symmetric Difference
Track Gap Detector
Union Creator

Add XYZ
Bearing
Ellipse
Event Volume Control
Extent Enricher
Field Grouper
GeoNames Lookup
Motion Calculator

Range Fan
Reverse Geocoder
Service Area Creator
Symbol Lookup
Track Idle Detector
Unit Converter
Visibility
Query Report

Esri Gallery

Add XYZ
Bearing
Ellipse
Event Volume Control
Extent Enricher
Field Grouper
GeoNames Lookup
Motion Calculator

Range Fan
Reverse Geocoder
Service Area Creator
Symbol Lookup
Track Idle Detector
Unit Converter
Visibility
Query Report
Extending GeoEvent

- Software Development Kit (SDK)

- You can create your own custom connectors and processors using the GeoEvent Software Development Kit (SDK)
Anatomy of a Processor

What makes up a Processor?

- Processor Service
  - Instantiates processor definition, processor and applies configuration

- Processor Definition
  - Defines processor metadata, properties and GeoEvent Definition(s)

- Processor
  - Implements processor lifecycle

- Configuration
  - Defines processor, it’s dependencies and metadata
Processor Service

- Create a Processor
- Create new Java class
- Extend GeoEventProcessorServiceBase class
- Creates processor definition
- Creates processor instance

```java
public class GeometryExtentEnricherService extends GeoEventProcessorServiceBase {
    private Messaging messaging;

    public GeometryExtentEnricherService() {
        definition = new GeometryExtentEnricherDefinition();
    }

    @Override
    public GeoEventProcessor create() throws ComponentException {
        GeometryExtentEnricher enricher = new GeometryExtentEnricher(definition);
        enricher.setMessaging(messaging);
        return enricher;
    }
}
```
Processor Definition
Properties and GeoEvent Definition(s) in Code

- Define collection of Properties that are required at run-time
- Provide Metadata about your Processor
- BundleLoggerFactory can log localized messages if so desired
  - Similar to log4j commons-logging

```java
public class GeometryExtentEnricherDefinition extends GeoEventProcessorDefinitionBase {
    private static final BundleLogger LOGGER = BundleLoggerFactory.getLogger(GeometryExtentEnricherDefinition.class);

    public GeometryExtentEnricherDefinition() {
        try {
            propertyDefinitions.put("geoEventDefinitionName",
                new PropertyDefinition("geoEventDefinitionName",
                    PropertyType.String, null,
                    "${com.esri.geoevent.processor.extent-enricher-processor.PROCESSOR_GED_NAME_LBL}",
                    "${com.esri.geoevent.processor.extent-enricher-processor.PROCESSOR_GED_NAME_DESC}",
                    true, false));
        }
```
Processor Definition

Metadata

Identification
1. Name
2. Domain
3. Version

```java
@Override
public String getName()
{
    return "GeometryExtentEnricher";
}

@Override
public String getDomain()
{
    return "com.esri.geoevent.processor";
}

@Override
public String getVersion()
{
    return "10.6.0";
}
```
Processor Definition

Metadata

Description

4. Label
5. Description
6. Contact Information

```java
@override
public String getLabel()
{
    return "${com.esri.geoevent.processor.extent-enricher-processor.PROCESSOR_LABEL}";
}

@override
public String getDescription()
{
    return "${com.esri.geoevent.processor.extent-enricher-processor.PROCESSOR_DESC}";
}

@override
public String getContactInfo()
{
    return "geoevent@esri.com";
}

public Map<String, GeoEventDefinition> getGeoEventDefinitions();
```
Processor Lifecycle

1. Creation
2. Initialization
3. Validation

```java
public class GeometryExtentEnricher extends GeoEventProcessorBase {
    // Constructor implementation

    protected GeometryExtentEnricher(GeoEventProcessorDefinition definition) throws ComponentException {
        super(definition);
    }
}
```
Processor Lifecycle

1. Creation
2. **Initialization**
3. Validation

```java
public void afterPropertiesSet()
{
    geoEventDefinitionName = Validator.compactWhiteSpaces(getProperty("geoEventDefinitionName").getValueAsString());
    addCenterPoint = Converter.convertToBoolean(getProperty("addCenterPoint").getValueAsString());
}
```
1. **Creation**
2. **Initialization**
3. **Validation**

```java
@Override
class Processor {
    public synchronized void validate() throws ValidationException {
        super.validate();
        if (geoEventDefinitionName.isEmpty()) {
            throw new ValidationException(LOGGER.translate("GED_EMPTY_ERROR"));
        }
    }
}
```
Processor Lifecycle

4. **Processing**

5. **Shutdown**

6. **Service Start / Stop**

```java
@Override
public GeoEvent process(GeoEvent geoEvent) throws Exception {
    GeoEvent augmentedGeoEvent = geoEvent;
    if (geoEvent != null && geoEventDefinitionManager != null) {
        GeoEventDefinition edOut = lookup(geoEvent.getGeoEventDefinition());
        augmentedGeoEvent = populateGeoEvent(geoEvent, edOut);
    }
    return augmentedGeoEvent;
}
```
4. **Processing**

5. **Shutdown**

6. **Service Start / Stop**
Processor Service
Configuration

• Declare processor service as a service implementing GeoEventProcessorService interface from GeoEvent Server SDK

```xml
<?xml version="1.0" encoding="UTF-8"?>
<blueprint xmlns="http://www.osgi.org/xmlns/blueprint/v1.0.0">
  <reference id="messagingService" interface="com.esri.ges.messaging.Messaging" timeout="1000"/>
  <bean id="geometryExtentEnricherServiceBean" class="com.esri.geoevent.processor.enricher.GeometryExtentEnricherService" activation="eager">
    <property name="bundleContext" ref="blueprintBundleContext"/>
    <property name="messaging" ref="messagingService"/>
  </bean>
  <service id="GeometryExtentEnricherService" ref="geometryExtentEnricherServiceBean" interface="com.esri.ges.processor.GeoEventProcessorService"/>
</blueprint>
```
Troubleshooting / Demo

What’s going wrong?

```java
public GeoEvent process(GeoEvent geoEvent) throws Exception {
    GeoEvent augmentedGeoEvent = geoEvent;
    if (geoEvent != null && geoEventDefinitionManager != null) {
        GeoEventDefinition edOut = lookup(geoEvent.getGeoEventDefinition());
        augmentedGeoEvent = populateGeoEvent(geoEvent, edOut);
    }
    return augmentedGeoEvent;
```
Debugging

- Open file C:\Program Files\ArcGIS\Server\GeoEvent\etc\ArcGISGeoEvent.cfg and un-comment lines (as Administrator!!)
  - Continue numbering scheme (actual number will depend on version of GeoEvent)
Troubleshooting

- Configure Debugging for Remote Java Application
Real-Time GIS: Creating Connectors using the GeoEvent SDK

Summary

- **Connectors** – Recipe for creating inputs/outputs
  - Transport – Moves raw data in/out of the GeoEvent Server
  - Adapter – Converts raw data to GeoEvents and back
  - Properties – Used to configure a transport, adapter, or processor for a specific use case. Allows the end user to configure component.

- **Processors**
  - GeoEvent allows you to develop and deploy custom processors
  - Can take advantage of Esri Java Geometry libraries
  - Extent Enricher available on GitHub: [https://github.com/Esri/extent-enricher-for-geoevent](https://github.com/Esri/extent-enricher-for-geoevent)

- **GeoEvent SDK** – Tools that help you to extend the GeoEvent capabilities and to build your own custom applications.
  - Samples
  - GeoEvent Developer Guide
Where to learn more?

Resources

• To learn more, visit the ‘Get Started’ area of the GeoEvent Server page:
  - http://links.esri.com/geoevent
    - Introduction
    - Big Data Store
    - Resiliency
    - Multiple-Machine Sites
    - Notifications
    - Stream Services
    - RSS, HTTP, Files
    - REST Admin API

• Ask questions on the GeoEvent Forum:
  - http://links.esri.com/geoevent-forum
Please Attend Our Other Sessions!

- GeoEvent Server: An Introduction  
  Tue, 2:30-3:30 pm, Primrose B

- GeoEvent Server: Applying Real-Time Analytics  
  Tue, 5:30pm-6:30 pm, Primrose A

- Real-Time and Big Data GIS: Best Practices  
  Wed, 10:30-11:30 am, Primrose B  
  Fri, 1:00-2:00 pm, Catalina/Madera

- ArcGIS and the Internet of Things (IoT)  
  Wed, 2:30-3:30 pm, Primrose B

- Real-Time and Big Data GIS: Leveraging the Spatiotemporal Big Data Store  
  Wed, 4:00-5:00 pm, Primrose A

- Developing Real-Time Web Apps with the ArcGIS API for JavaScript  
  Tue, 9:00-9:30 am, Demo Theater 1: Oasis 1-2

- Real-Time GIS: Road Ahead  
  Thu, 4:00-5:00 pm, Pasadena/Sierra/Ventura

- GeoEvent Server: Creating Connectors and Processors Using the GeoEvent SDK  
  Fri, 8:30-9:30 am, Mesquite B
Please Take Our Survey!

Download the Esri Events app and find your event

Select the session you attended

Scroll down to the “Feedback” section

Complete Answers, add a Comment, and Select “Submit”
Esri.com Proceedings page

http://proceedings.esri.com

Where users can find PDF’ed slides after DevSummit