Do you have real-time, streaming-data, event-driven workflows? Tapis Streams API can help!

Smruti Padhy*, Ph.D.
Anagha Jamthe, Sean B. Cleveland, Jack A. Smith, Joe Stubbs, Christian Garcia, Michael Packard, Julia Looney, Steve Terry, Richard Cardone, Maytal Dahan, Gwen A. Jacobs

*Cloud and Interactive Computing
Texas Advanced Computing Center
University of Texas, Austin
Acknowledgements

- National Science Foundation - Office of Advanced Cyberinfrastructure
  Collaborative Proposal: Frameworks: Project Tapis: Next Generation Software for Distributed Research (Award #1931439, #1931575)
  SI2-SSE: Abaco - Flexible, Scalable, and Usable Functions-As-A-Service via the Actor Model (Award #1740288)
- Tapis and Abaco Team
- Early Adopters
TALK OUTLINE

- Challenges in Computational Research
- The Tapis Project Overview
- Motivation
- Tapis v3 Streams API Overview
- High-level Architecture
- Streams API Design
- An Example Science Use Case
- Experimental Setup and Evaluation
- How Tapis Streams API can help you?
- Upcoming Features
Challenges of Computational Research

Instruments and Sensors
- Sequencers
- Telescopes
- Shake Tables
- Wind Tunnels
- Lasers

Data Processing
- Quality Assurance
- ETL scripts
- Machine Learning

Model Simulation
- OpenMP
- MPI
- CUDA

Sharing & Publishing
- Permissions
- History
- Provenance
- Reproducibility
The Tapis Project

NSF funded, Web-based Computing Framework To Support Research

- Generally - A framework to support computational research, in any domain, that enables investigators to leverage computing resources across various institutions.
- More Technically - A set of hosted, multi-datacenter REST APIs, together with languages SDKs and CLI for securely managing data and executing code on HPC, HTC and cloud systems.
Higher-Level Tapis Objectives

Tapis Gives Researchers The Ability to...

- **Manage Data and Execute workflows** - Tapis enables secure execution of workflows that can span geographically distributed resources. It manages access to data and results by attaching permissions to any resource in the API.

- **Track your analysis provenance** - Tapis records your input and output data along with application used and settings - so you know what you have done every time.

- **Reproduce your analysis** - Tapis records all your inputs/outputs/parameters etc so you can re-run an analysis.

- **Share your data, workflows/applications, computational resources with collaborators or your lab** - Tapis enables sharing with access controls for all your data/resources/applications within Tapis.

Without having to install or support a complicated stack of technology
Tapis: Overview

Primary Capabilities - Version Three (v3)

- Systems, Files, Apps, Jobs - *Interact with data and execute code on HPC and HTC systems.*
- Metadata - *Manage large collections of document objects.*
- Abaco Functions - *Execute containerized functions in response to messages.*
- Authentication Module - OAuth2/OIDC compliant front-end.
- **Containerized Apps** - *First class support for containers.*
- **Streams** - *Store and process sensor data in real-time.*
- **Security Kernel** - *Decentralized secrets store and authorization subsystem.*

*New in v3*
Motivation
Need for Support for Real-time, Streaming-data, Event-driven Workflows

- Rise of IoT devices, instruments, and sensors to observe and measure everything
  - increase in time-series data
  - increase in demand for support for storing, processing and analyzing time-series data
- Many science use cases based on monitoring require
  - ongoing data processing or special processing/modeling as it arrives.
  - notification of anomalous or special events as identified from the above step.
- Many existing computational infrastructure and system are built around batch computation.
Need for Support for Real-time, Streaming-data, Event-driven Workflows

- Ready off-the-shelf solutions to handling streaming data
  - Elasticsearch, Splunk, Apache Flume, Prometheus, Apache Kafka, Apache Storm, Apache Fink, Apache Spark Streaming
  - Amazon Timestream, Azure Time Series Insights, ArcGIS Analytics for IoT
- Geoscience time-series datastores:
  - Hydroserver
  - Virtual observatory and ecological informatics system (VOEIS)
  - Cloud-Hosted Real-time Data Service (CHORDS)

Meets use cases for data centers and events but lack integrated collaborative features and spatial support or are too expensive or complex to manage for many researchers.
Tapis v3 Streams API Overview
Tapis v3 Streams API

- Hosted Real-time, Streaming Data Storage/Processing & Events Service
- Stream data from geo-distributed sensors
- Extends CHORDS (NSF Earth Cube) data model (sites, instruments, variables)
- Search/slice data using geospatial and temporal indexes
Tapis Streams API Core Features

- Data Ingestion - Support for time-series float and string data from devices and instruments.
- Data Annotation - Integrated extendable annotation for data curation and discovery.
- Data Storage - Hosted two-tiered storage for fast windowed processing and long-term storage within InfluxDB and MongoDB.
- Data Archival - Support archival and data replication within external CHORDS instances and flat file formats.
- Data Retrieval (with temporal and spatial support) - Search APIs that enable spatial extent and temporal constraints for advanced queries.
Aim: Support for Real-time, Streaming Data, Event-Driven Workflow

- Defining evaluation functions on multiple incoming data streams
- Evaluation of multiple measurement variables to trigger analysis workflows
- Integration with the Tapis Abaco (Function-as-a-service)
- Integration with computational Applications and Jobs services
- Collaboration Support - Role-based access allowing multiple users to share and manage both data and workflows.
Tapis v3 Streams API Architecture and Background
High-Level Architecture of Tapis v3 Streams API
Hierarchy

Projects {
  Sites {
    Instruments {
      Variables
    }
  }
}
CHORDS

- Cloud-hosted real-time data service infrastructure that provides a graphical interface for acquiring, storing, and analyzing data streams via cloud services and the Internet.
- It leverages InfluxDB and MySQL databases.
- Basic Data Models - Sites, Instruments, Variables, Measurements.
- Metadata stored in MySQL.
- Measurements are written to InfluxDB.
- Scientists and Analysts can easily fetch data in real-time from CHORDS portal, delivered directly to browsers, programs, and mobile apps.
Measurement Data

- Stored in InfluxDB
- Currently supporting Float data types
- Written in the context of an Instrument & Variables
  - Multiple variable values for a single ISO-8601 timestamp
Kapacitor

- Native Real-time data processing engine for InfluxDB 1.x
- Process batch and real-time streaming data from InfluxDB
- Defines monitoring and processing task to perform on the incoming data via TICKScript
- Enables ways to write TICKscript template task, define a task using a template and set up trigger and alert actions
- Default choice towards achieving real-time data-event-driven workflows
Abaco Functions-as-a-Service

- Define computational primitives called “actors” using a Docker image.
- Send messages to actors using HTTP - Abaco queues an execution for each message.
- Abaco asynchronously launches a container for each message; the message data are injected into the container.
- Actors can store results and logs, and Abaco measures resources used.
- Platform scales to thousands of concurrent executions across hundreds of nodes.
Tapis v3 Streams API Design
## Streams API Resources

<table>
<thead>
<tr>
<th>Resource Name</th>
<th>Method allowed</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>GET, POST, PUT, DEL</td>
<td>Logical grouping of sites</td>
</tr>
<tr>
<td>Site</td>
<td>GET, POST, PUT, DEL</td>
<td>Geographical location which may host one or more instruments</td>
</tr>
<tr>
<td>Instrument</td>
<td>GET, POST, PUT, DEL</td>
<td>A source of related measurements</td>
</tr>
<tr>
<td>Variable</td>
<td>GET, POST, PUT, DEL</td>
<td>A particular measurement made by an instrument</td>
</tr>
<tr>
<td>Measurement</td>
<td>GET, POST</td>
<td>A single observation of a for an instrument</td>
</tr>
<tr>
<td>Channel</td>
<td>GET, POST, PUT, DEL</td>
<td>Pre-processing of measurements to trigger alert or continuous query</td>
</tr>
<tr>
<td>Template</td>
<td>GET, POST, PUT, DEL</td>
<td>Template task script to define and reuse in a channel definition</td>
</tr>
</tbody>
</table>
Streaming Data-Event-Driven Workflow

Data Ingestion
- Measurement 1: Instrument A:
  - Var1: 2.0
  - Var2: 10.5
  - DateTime: 2021-02-01T01:20:00Z
- Measurement 2: Instrument B:
  - Var1: 0.0
  - Var2: 90.2
  - DateTime: 2021-02-01T01:20:00Z

Tapis Streams API

Data Evaluation
- Channel 1: If Instrument A:
  - Var1 < 5.0
  - Trigger Action 1
- Channel 2: If Instrument B:
  - Var2 < 75.0
  - Trigger Action 2
- Channel 3: If Instrument B:
  - Var1 < 3.0 AND Instrument A:
    - Var1 < 3.0
    - Trigger Action 3

Trigger Alerts
- Alert Channel 1
- Alert Channel 2
- Alert Channel 3

Execute Actions
- Action 1
  - Execute with Measurement 1 data
- Action 3
  - Execute with Measurement 1 and Measurement 2 data
Template

- Task - A work to be performed on a streaming dataset.
- Template (or Template Task) - A re-usable definition that represents a task.
- Structure of a Task
  - Monitoring of incoming measurements data
  - Aggregating values
  - Raising alerts when the measurement values satisfy specified boolean conditions
  - Sending the critical data that triggered the alert to perform specified actions.
- Templates are created before defining a channel. There will be predefined templates for users.
- Enables reuse of the template script with different conditions and alerts actions so multiple channels can be defined.
An Example Template

- The Streams API wraps Kapacitor's templates API and tasks API
- A template is a TICKScript defining a task with variables that are set during the channel definition
- Leverages Kapacitor's TICKScript lambda expression representation

```javascript
var measurement string
var channel_id string
var crit1 lambda
var crit2 lambda
var crit3 lambda

var vari = stream
|from()
.measurement(measurement)
.where(lambda:crit1)

var var2 = stream
|from()
.measurement(measurement)
.where(lambda:crit2)

|join(var2)
.as('var1','var2')
.tolerance(1ms)
.fill(0.0)
.streamName('multi')

|alert()
.id(channel_id + ' {{ .Name }}/{{ .Group }}/\{ .TaskName }\')
.crit(lambda:crit3)
.noRecoveries()
.message('{{ .ID }} is {{ .Level }} at time: {{ .Time }} as value exceeded the threshold')
.details('')
.post()
.endpoint('api-alert')
.captureResponse()
|httpOut('msg')
```

- Declaration of TICKScript variables
- Setting up TICKScript variables to filter data streams
- Aggregation of data
- Alerts Definitions
Channels Design
Channels : Real-time Data Processing & Events

- An abstraction for pre-processing real-time streaming data to trigger alerts based on a function of the variables and initiate analysis on the data.
- A function of the variables is a boolean criteria/condition on the variables' values.
- When the evaluation is True an Alert is issued and additional processes can be initiated such as Abaco actors (Containerized script)
- Template task need to be created prior to channel creation
- A channel defines a task for the Kapacitor using a template task and measurement variables with boolean conditions and a set of actions.
Channel Definition Example

```json
{channel_id='tapis-demo-channel',
 channel_name='demo_channel',
 template_id='demo_tapis_two_variable',
 triggers_with_actions=[{
   "inst_ids":["instrument1"],
   "condition": ["AND", {
                         "key": "instrument1.var1",  "operator": ">",  "val":90},
                         {
                         "key": "instrument1.var2",  "operator": "<="  ,  "val":150}]
   "action":{"method":"ACTOR",  
   "actor_id" :actor_id,
   "message":"Instrument: instrument1 exceeded threshold"
   "abaco_base_url":"https://api.tacc.utexas.edu"
   }]}
}
Streams API’s architecture to support event-driven workflow

When the boolean condition in the channel definition is true, Set of Actions performed

- **Alerts** as event are generated
- Pre-processing of the alert data
- Initiating processing of measurements using Tapis Abaco function, thereby enabling real-time streaming data event-driven workflow
An Example Science Use Case
Researchers at Marshall University are monitoring water quality of streams in Appalachia.

3 NSF-funded projects: Appalachian Freshwater Initiative (AFI), Sensing and Educating the Nexus to Sustain Ecosystems (SENSE), Extensible Geospatial Data Framework towards FAIR Science (GeoEDF).

It involves:

- Collection and sharing of water quality data using EPA’s WQP/WQX API and local staging service call WQBase for discrete data.
- CHORDS Instance for continuously-monitored sensor data.
- With the need for on-the-fly processing, potentially on remote HPC.
A Real-Time Event-Driven Workflow Example

- Envision to get benefited from real-time analysis
- Goal of integrating water quality science gateways with Streams API is to
  - detect anomalies in the measured data,
  - correct for calibration errors
  - generate alerts when certain thresholds are exceeded
  - computing rolling statistics or combine with other data sources,
  - use machine learning and image processing to recognize conditions favorable to harmful algae bloom formation,
  - and push summary data to WQBASE for Staging, pending further vetting and analysis before publishing to WQP via WQX.
Experimental Setup and Evaluation
Experimental Setup

- Water quality event-driven workflow Example
- Real-time datasets from two sites from the Ohio River
- Each site hosts two instruments (YSI EXO2 Sondes)
- Each instrument has multi-sensor probes monitoring different properties for water quality study
- Data obtained every 15 mins for 30 days

<table>
<thead>
<tr>
<th>Variable</th>
<th>Instrument 1</th>
<th>Instrument 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Battery (V)</td>
<td>8.9 - 15</td>
<td>12.6 - 12.9</td>
</tr>
<tr>
<td>Temperature (C)</td>
<td>1.7 - 7.06</td>
<td>2.42 - 7.47</td>
</tr>
<tr>
<td>Sp Cond (μS/cm)</td>
<td>195 - 319</td>
<td>247 - 357</td>
</tr>
<tr>
<td>PH</td>
<td>7.4 - 8.02</td>
<td>7.29 - 7.93</td>
</tr>
<tr>
<td>Turbidity (ntu)</td>
<td>6.9 - 761</td>
<td>11.52 - 264.21</td>
</tr>
<tr>
<td>Odo sat. (%)</td>
<td>92 - 112</td>
<td>93.72 - 100.61</td>
</tr>
<tr>
<td>Odo (mg/L)</td>
<td>11.28 - 13.6</td>
<td>11.42 - 13.49</td>
</tr>
<tr>
<td>Chlorophyll (μg/L)</td>
<td>1.27 - 6.28</td>
<td>1.11 - 6.55</td>
</tr>
<tr>
<td>Chlorophyll (RFU)</td>
<td>0.3 - 1.5</td>
<td>0.3 - 2.6</td>
</tr>
<tr>
<td>BGA-Phycocyanin (μg/L)</td>
<td>0.43 - 2.55</td>
<td>0.29 - 0.86</td>
</tr>
<tr>
<td>BGA-Phycocyanin (RFU)</td>
<td>0.4 - 2.6</td>
<td>0.3 - 0.8</td>
</tr>
<tr>
<td>Depth (m)</td>
<td>0.727 - 6.265</td>
<td>0.263 - 2.34</td>
</tr>
<tr>
<td>Wiper Pos. (V)</td>
<td>1.18 - 1.21</td>
<td>1.18 - 1.23</td>
</tr>
<tr>
<td>Cable Power. (V)</td>
<td>8.9 - 14</td>
<td>12.1 - 13.7</td>
</tr>
<tr>
<td>PAR (μmol/s/m2)</td>
<td>7.4 - 1472</td>
<td>7.4 - 1468</td>
</tr>
</tbody>
</table>
Simulation Results

- Created three channels for each instrument
- Used one, two and three variables templates for channels creation respectively
- One variable condition: Temp>9
- Two variable condition: Temp>9 and pH <5
- Three variable condition: Temp>9 AND pH<5 OR Turb.>500
- We deliberately injected data that could satisfy the boolean condition and raise alerts

<table>
<thead>
<tr>
<th>Create Measurement</th>
<th>Num. of Conditions</th>
<th>Inst.1: alerts generated/expected</th>
<th>Inst.2: alerts generated/expected</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1s</td>
<td>One</td>
<td>13/13</td>
<td>13/13</td>
</tr>
<tr>
<td>0.5s</td>
<td>One</td>
<td>12/12</td>
<td>11/11</td>
</tr>
<tr>
<td>1s</td>
<td>One</td>
<td>10/10</td>
<td>12/12</td>
</tr>
<tr>
<td>0.1s</td>
<td>Two</td>
<td>3/3</td>
<td>5/5</td>
</tr>
<tr>
<td>0.5s</td>
<td>Two</td>
<td>2/2</td>
<td>5/5</td>
</tr>
<tr>
<td>1s</td>
<td>Two</td>
<td>6/6</td>
<td>1/1</td>
</tr>
<tr>
<td>0.1s</td>
<td>Three</td>
<td>6/6</td>
<td>3/3</td>
</tr>
<tr>
<td>0.5s</td>
<td>Three</td>
<td>4/4</td>
<td>3/3</td>
</tr>
<tr>
<td>1s</td>
<td>Three</td>
<td>3/3</td>
<td>3/3</td>
</tr>
</tbody>
</table>
How Tapis Streams API can help you?

- Steps for building event-driven workflow with Streams API
  - Real-time sensor float type data
  - Create an Abaco actor
  - Create Stream resources - Project, site, instrument, variables,
  - Create template with the monitoring task
  - Create Channel using the template and the abaco actor
  - Post measurements to Channels

- Tools available building such workflow -
  - tapipy - tapis python sdk
  - tapis-jupyter-notebook
Upcoming Features

- Advanced Search API for geospatial queries
- Ontology Support for Metadata
- More Channel endpoint (Jobs, 3rd Party APIs etc)
- Additional Data Type support
- UI for managing Streams Data & Events

Other Input From Early Adopters and Community Welcomed!
THANKS!

Connect with Us
https://tapis-project.org

More information:
- Developer Docs: https://tacc-cloud.readthedocs.io
- OpenAPI v3 Live docs: https://tapis-project.github.io/live-docs/
Thanks!