



**Department of Computer Science and Engineering (Data Science)**

**Subject: Big Data Engineering (DJ19DSL604)**

**AY: 2022-23**

**Experiment 3**

**(Messaging Services)**

**NAME: Kresha Shah**

**SAP ID: 60009220080**

**Aim:** Implement messaging system using AMPS

**Theory:**

Communication, in general, is the process of transferring data from a source to a destination by using any of the available modes such as audio, video, text or even signals, etc. This communication may be straightforward between one sender and one receiver or it may include multiple senders and receivers. On the basis of the number of senders and receivers involved, a communication can either be "Point-to-Point" or "Multi-point".

**Point-to-Point Communication:**

In telecommunications, a point-to-point connection is a communications link between two communication endpoints or nodes. A telephone call is an example of this, in which two phones are linked, and what one caller says can only be heard by the other.

A "point-to-multipoint" or broadcast link, on the other hand, allows multiple nodes to receive information sent by a single node. Leased lines and microwave radio relays are also examples of point-to-point connections.



### Department of Computer Science and Engineering (Data Science)

In a point-to-point communication, there will be a transmitter and a receiver connected together with a suitable connection. The capacity of the connecting channel remains unchanged throughout the communication.

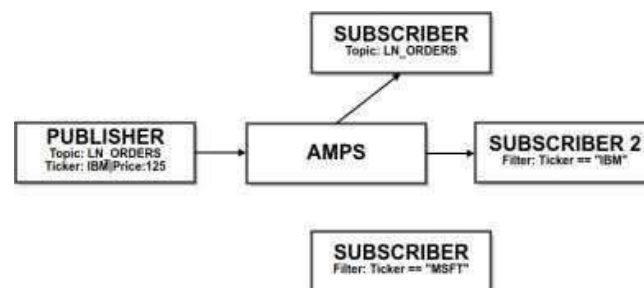
### Multi-point Communication

**In telecommunications, point-to-multipoint communication (P2MP, PTMP, or PMP) is a form of one-to-many communication that allows numerous routes to be established from a single site to several locations.**

The usage of gigahertz radio frequencies for point-to-multipoint telecommunications is common in wireless Internet and IP telephony. P2MP systems have been created with and without a return channel from the numerous receivers. The system employs a kind of time-division multiplexing to enable the return channel traffic, which is transmitted from a central antenna to numerous receiving antennas.

**AMPS from 60East Technologies** is a fast messaging engine that supports both publish-subscribe messaging and queuing. It is not just a simple messaging engine it is also packed with some powerful feature like high availability, historical replay, aggregation and analytics, content filtering and continuous query, last value caching, focus tracking, and more.

### AMPS PUB-SUB Model



This is same as any other pub-sub model wherein a publisher publishes the message to a topic and subscriber subscribes to a topic and reads the message from it. But amps also provide some

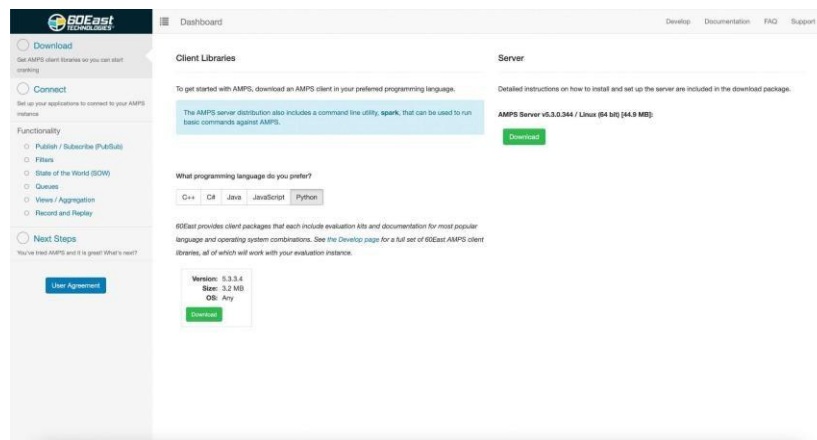


### Department of Computer Science and Engineering (Data Science)

powerfully features like we can do **content filtering** where we can subscribe to a topic with a filter and with the help of that we can get only the filtered messages from AMPS server.

#### Setting up AMPS:

- Visit <https://www.crankuptheamps.com/evaluate/>
- Sign up with any email ID
- You'll receive an evaluation kit of AMPS in your email.
- Download the AMPS Server
- Download the Client Libraries (as per programming language preference)



- After downloading the server binaries and client, click on Connect



## Department of Computer Science and Engineering (Data Science)

**SOEast TECHNOLOGIES** Dashboard

**Download**  
Get AMPS client libraries so you can start cranking

**Connect**  
Set up your applications to connect to your AMPS instance

**Functionality**

- ☐ Publish / Subscribe (Pub/Sub)
- ☐ Filters
- ☐ State of the World (SOW)
- ☐ Queues
- ☐ Views / Aggregation
- ☐ Record and Replay

**Next Steps**  
You've tried AMPS and it's great! What's next?

[User Agreement](#)

**Configure AMPS**

C# C++ **Java** JavaScript Python

**To get started with AMPS:**

**Server**

To install the Linux distribution of AMPS and start the evaluation server:

1. Extract the Linux distribution and the client package.
2. Make a config directory in the AMPS distribution directory, for example:  
`mkdir ~/amps_dir/config`
3. Copy the sample.xml file from the evaluation kit into the config directory, for example:  
`cp ~/java_client_package/samples/sample.xml ~/amps_dir/config/sample.xml`
4. Change directories to the AMPS distribution directory and start AMPS with the sample config file. For example:  
`cd ~/amps_dir/  
./bin/ampServer -./config/sample.xml`

**Client**

1. Open a command line window:
  - a. Navigate to the directory where you extracted the client package
  - b. Navigate to **Java Client Package / samples**
2. Build the sample program.

- On the right side pane, you'll see the steps to configure AMPS Server and Client.
- Once configured, head over to the Functionality tab (below Connect)
- Test your AMPS configuration by publishing / subscribing to a topic.

**SOEast TECHNOLOGIES** Publish Messages / Subscribe for Messages

**Download**  
Get AMPS client libraries so you can start cranking

**Connect**  
Set up your applications to connect to your AMPS instance

**Functionality**

- ☒ Publish / Subscribe (Pub/Sub)
- ☐ Filters
- ☐ State of the World (SOW)
- ☐ Queues
- ☐ Views / Aggregation
- ☐ Record and Replay

**Next Steps**  
You've tried AMPS and it's great! What's next?

[User Agreement](#)

**AMPS Publish and Subscribe**

Low-latency publish and subscribe messaging is at the heart of AMPS. The other features of AMPS build on this foundation.

[Learn More >](#)

**Subscribe to a topic**

This sample shows how to subscribe to a topic. Notice that, with simple publish/subscribe messaging, messages will only be received on the topic when messages are published.

C# C++ **Java** JavaScript Python **Spark**

```
1 from AMPS import *
2
3 # Construct a client object
4 c = Client("subscriber")
5
6 # Connect and login
7 c.connect("tcp://127.0.0.1:9007/amps/json")
8 c.login()
9
10 # Subscribe
```

AMPS is now ready to be explored.

### Lab Assignment:



### Department of Computer Science and Engineering (Data Science)

1. Setup AMPS server in the lab and all students will connect to it.
2. Write basic JSON messages (having flat structures), publish, and consume using AMPS.
3. Write nested JSON messages (having 5-6 levels of hierarchy), publish and subscribe using AMPS.
4. Apply Content Filtering, subscribing to relevant messages only.

```
cs-ds@kmaster: ~/amps_dir
cs-ds@kmaster:~$ mkdir amps_dir
cs-ds@kmaster:~$ mkdir config
cs-ds@kmaster:~$ cd amps_dir
cs-ds@kmaster:~/amps_dir$ mkdir config
cs-ds@kmaster:~/amps_dir$ cd config
cs-ds@kmaster:~/amps_dir/config$ cp ~/amps-python-client-5.3.4.1/samples/sample.xml
~/amps_dir/config/sample.xml
cs-ds@kmaster:~/amps_dir/config$ cd amps_dir
bash: cd: amps_dir: No such file or directory
cs-ds@kmaster:~/amps_dir/config$ cd ~/amps_dir/
cs-ds@kmaster:~/amps_dir$
```



Shri Vile Parle Kelavani Mandal's

**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**

(Autonomous College Affiliated to the University of Mumbai)

NAAC Accredited with "A" Grade (CGPA : 3.18)



### Department of Computer Science and Engineering (Data Science)

```
cs-ds@kmaster: ~/AMPS-5.3.3.150-Release-Linux
cs-ds@kmaster:~/AMPS-5.3.3.150-Release-Linux$ ./bin/ampServer
AMPS 5.3.3.150.084006.7b89fe3 - Copyright (c) 2006-2021 60East Technologies Inc.
(Built: 2024-01-30T17:55:29Z)
For all support questions: support@crankuptheamps.com

Usage:

  ampServer [--help | --version | --sample-config]
  or
  ampServer [--verify-config | --daemon] <config>

Options:

  --help           : prints this help message
  --version        : print version and exit
  --sample-config  : print a configuration template to standard out
  --verify-config  : verify configuration file and exit
  --dump-config    : output configuration file contents to standard out
  --daemon         : run as a background daemon
  config          : configuration file to load

Examples:

% ./ampServer orders.xml
```



Shri Vile Parle Kelavani Mandal's

**DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING**

(Autonomous College Affiliated to the University of Mumbai)

NAAC Accredited with "A" Grade (CGPA : 3.18)



### Department of Computer Science and Engineering (Data Science)

```
cs-ds@kmaster: ~/AMPS-5.3.3.150-Release-Linux
AMPS will print a sample configuration that is redirected to the file
'template.xml'.

Error: AMPS configuration file does not exist: /home/cs-ds/AMPS-5.3.3.150-Release-Linux/config/sample.xml
cs-ds@kmaster:~/AMPS-5.3.3.150-Release-Linux$ ./bin/ampServer ./config/sample.xml
AMPS 5.3.3.150.084006.7b89fe3 - Copyright (c) 2006-2021 60East Technologies Inc.
(Built: 2024-01-30T17:55:29Z)
For all support questions: support@crankuptheamps.com

2024-02-14T11:28:39.5084470+05:30 [1] info: 00-0015 AMPS initialization completed (3 seconds).
^C
interrupted: AMPS is shutting down.
cs-ds@kmaster:~/AMPS-5.3.3.150-Release-Linux$ ./bin/ampServer ./config/sample.xml
AMPS 5.3.3.150.084006.7b89fe3 - Copyright (c) 2006-2021 60East Technologies Inc.
(Built: 2024-01-30T17:55:29Z)
For all support questions: support@crankuptheamps.com

2024-02-14T11:33:04.9989540+05:30 [1] info: 00-0015 AMPS initialization completed (3 seconds).
```





**Department of Computer Science and Engineering (Data Science)**

```
cs-ds@kmaster: ~/amps-python-client-5.3.4.1
cs-ds@kmaster:~/amps-python-client-5.3.4.1$ sudo python3 setup.py build install
[sudo] password for cs-ds:
/home/cs-ds/amps-python-client-5.3.4.1/setup_amps.py:30: DeprecationWarning: The
distutils package is deprecated and slated for removal in Python 3.12. Use setu
ptools or check PEP 632 for potential alternatives
  from distutils.core import setup
/home/cs-ds/amps-python-client-5.3.4.1/setup_amps.py:32: DeprecationWarning: The
distutils.sysconfig module is deprecated, use sysconfig instead
  from distutils.sysconfig import get_config_var
running build
running build_clib
building 'amps' library
creating src/cpp/lib
creating src/cpp/lib/src
creating src/cpp/lib/src/cpp
creating src/cpp/lib/src/cpp/src
x86_64-linux-gnu-gcc -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O2 -
Wall -g -fstack-protector-strong -Wformat -Werror=format-security -g -fwrapv -O2
 -std=gnu11 -Wall -Wdate-time -D_FORTIFY_SOURCE=2 -fPIC -Isrc/cpp/src -Isrc/cpp/
include -Iinclude -c src/cpp/src/client.c -o src/cpp/lib/src/cpp/src/client.o
x86_64-linux-gnu-gcc -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O2 -
Wall -g -fstack-protector-strong -Wformat -Werror=format-security -g -fwrapv -O2
 -std=gnu11 -Wall -Wdate-time -D_FORTIFY_SOURCE=2 -fPIC -Isrc/cpp/src -Isrc/cpp/
include -Iinclude -c src/cpp/src/transport.c -o src/cpp/lib/src/cpp/src/transpo
```

```
cs-ds@kmaster: ~/amps-python-client-5.3.4.1/samples
cs-ds@kmaster:~/amps-python-client-5.3.4.1/samples$ python3 AMPSConsolePublisher
.py &
[2] 15810
cs-ds@kmaster:~/amps-python-client-5.3.4.1/samples$ { "hi" : "Hello, world!"}
^C
[2]+  Done                  python3 AMPSConsolePublisher.py
cs-ds@kmaster:~/amps-python-client-5.3.4.1/samples$ python3 AMPSConsoleSubscribe
r.py &
[2] 16034
cs-ds@kmaster:~/amps-python-client-5.3.4.1/samples$ python3 AMPSSOWConsolePublis
her.py &
[3] 16221
cs-ds@kmaster:~/amps-python-client-5.3.4.1/samples$ python3 AMPSSOWConsoleSubscr
iber.py &
[4] 16340
[3]  Done                  python3 AMPSSOWConsolePublisher.py
```





**Department of Computer Science and Engineering (Data Science)**

```
cs-ds@kmaster: ~/amps-python-client-5.3.4.1/samples
[3] Done python3 AMPSSOWConsolePublisher.py
cs-ds@kmaster:~/amps-python-client-5.3.4.1/samples$ Receiving messages from the
SOW.
{ "text":"Hello, world!", "messageNumber": 0 }
{ "text":"Hello, world!", "messageNumber": 1 }
{ "text":"Hello, world!", "messageNumber": 2 }
{ "text":"Hello, world!", "messageNumber": 3 }
{ "text":"Hello, world!", "messageNumber": 4 }
{ "text":"This is new information", "messageNumber":5 }
{ "text":"Hello, world!", "messageNumber": 6 }
{ "text":"Hello, world!", "messageNumber": 7 }
{ "text":"Hello, world!", "messageNumber": 8 }
{ "text":"Hello, world!", "messageNumber": 9 }
Done receiving messages from the SOW.
^C
[4]+ Done python3 AMPSSOWConsoleSubscriber.py
cs-ds@kmaster:~/amps-python-client-5.3.4.1/samples$ python3 AMPSSOWandSubscribec
onsoleSubscriber.py &
[3] 16397
```

```
cs-ds@kmaster: ~/amps-python-client-5.3.4.1/samples
[4]+ Done python3 AMPSSOWConsoleSubscriber.py
cs-ds@kmaster:~/amps-python-client-5.3.4.1/samples$ python3 AMPSSOWandSubscribec
onsoleSubscriber.py &
[3] 16397
cs-ds@kmaster:~/amps-python-client-5.3.4.1/samples$ Receiving messages from the
SOW.
(sow) : b'{ "text":"Hello, world!", "messageNumber": 0 }'
(sow) : b'{ "text":"Hello, world!", "messageNumber": 1 }'
(sow) : b'{ "text":"Hello, world!", "messageNumber": 2 }'
(sow) : b'{ "text":"Hello, world!", "messageNumber": 3 }'
(sow) : b'{ "text":"Hello, world!", "messageNumber": 4 }'
(sow) : b'{ "text":"This is new information", "messageNumber":5 }'
(sow) : b'{ "text":"Hello, world!", "messageNumber": 6 }'
(sow) : b'{ "text":"Hello, world!", "messageNumber": 7 }'
(sow) : b'{ "text":"Hello, world!", "messageNumber": 8 }'
(sow) : b'{ "text":"Hello, world!", "messageNumber": 9 }'
Done receiving messages from the SOW.
^C
cs-ds@kmaster:~/amps-python-client-5.3.4.1/samples$ python3 AMPSSOWConsolePublis
her.py &
[4] 16499
```



### Department of Computer Science and Engineering (Data Science)

```
cs-ds@kmaster:~/amps-python-client-5.3.4.1/samples$ (publish) : b'{ "text":"Hello, world!", "messageNumber": 0 }'  
(publish) : b'{ "text":"Hello, world!", "messageNumber": 1 }'  
(publish) : b'{ "text":"Hello, world!", "messageNumber": 2 }'  
(publish) : b'{ "text":"Hello, world!", "messageNumber": 3 }'  
(publish) : b'{ "text":"Hello, world!", "messageNumber": 4 }'  
(publish) : b'{ "text":"Hello, world!", "messageNumber": 5 }'  
(publish) : b'{ "text":"Hello, world!", "messageNumber": 6 }'  
(publish) : b'{ "text":"Hello, world!", "messageNumber": 7 }'  
(publish) : b'{ "text":"Hello, world!", "messageNumber": 8 }'  
(publish) : b'{ "text":"Hello, world!", "messageNumber": 9 }'  
(publish) : b'{ "text":"This is new information", "messageNumber":5 }'  
^C  
[4]+ Done python3 AMPSSOWConsolePublisher.py  
cs-ds@kmaster:~/amps-python-client-5.3.4.1/samples$
```

filter.py	publish.py	subscribe.py
<pre>1 c = Client("subscriber") 2 3 # Connect and login 4 c.connect("tcp://127.0.0.1:9007/amps/json") 5 c.login() 6 7 # Subscribe with a filter 8 for m in c.subscribe("test", "/details/items/description LIKE   'kitten' "): 9     print "Received message for topic: %s : %s" % (m.get_topic(), 10    m.get_data())</pre>	<pre>1 from AMPS import * 2 3 # Construct a client object 4 c = Client("publisher") 5 6 # Connect and login 7 c.connect("tcp://127.0.0.1:9007/amps/json") 8 c.login() 9 10 # Publish 11 c.publish("test", "{ "message" : "Hello, world!" }")</pre>	<pre>1 from AMPS import * 2 3 # Construct a client object 4 c = Client("subscriber") 5 6 # Connect and login 7 c.connect("tcp://127.0.0.1:9007/amps/json") 8 c.login() 9 10 # Subscribe 11 for m in c.subscribe("test"): 12     print "Received message: %s" % (m.get_data())</pre>



## Department of Computer Science and Engineering (Data Science)

```
publish.py - amps - Visual Studio Code

c:\students\csestudent-HP-ProOne-680-66-22-All-in-One-PC\Downloads\amps5 /usr/bin/env /bin/python3 /home/cse-student/.vscode/extensions/ms-python.python-2023.4.1/p
pythonFiles/lib/python/adapter/.vscode/.debugpy/launcher 49805 -- /home/cse-student/Downloads/amps/client.py

Received message: {
  "2017-12-31": {
    "Junior": {
      "Electronics": {
        "A": {
          "sales": -0.3947134370181142
        },
        "B": {
          "sales": -0.9873530754403204
        },
        "C": {
          "sales": -1.1182598058984508
        }
      },
      "Household": {
        "A": {
          "sales": -1.1211850078998677
        },
        "B": {
          "sales": 2.8330914483907847
        },
        "C": {
          "sales": 3.94762379718749
        }
      }
    },
    "Senior": {
      "Electronics": {
        "A": {
          "sales": 1.4528493451404196
        },
        "B": {
          "sales": -2.327732345261005
        },
        "C": {
          "sales": -2.8040263791743922
        }
      },
      "Household": {
        "A": {
          "sales": 3.8972919329279663
        },
        "B": {
          "sales": 9.884565742562392
        },
        "C": {
          "sales": 2.9359830722457576
        }
      }
    }
  },
  "2018-01-31": {
    "Junior": {
      "Electronics": {
        "A": {
          "sales": -1.358380149125217
        }
      }
    }
  }
}
```

```
publisher.py - AMPS_Lab - Visual Studio Code

amps-python-client-5.3.3.4 > publisher.py ...
6 c.connect((tcp://127.0.0.1:1883)/amps/json)
7 c.logon()
10
11 # Publish JSON data
12 data = {
13   "invoice": 53498,
14   "customerID": 1983,
15   "details": {
16     "discountCode": "1347",
17     "items": [
18       {
19         "sku": 3317,
20         "qty": 460,
21         "description": "action figure : purple : fox"
22       },
23       {
24         "sku": 5098,
25         "qty": 283,
26         "description": "shoes : beige : kitten"
27       }
28     ]
29   }
30 }
31
32 # Publish
33 # c.publish("test", '{"message": "Hello, world!"}')
34
35 c.publish("test", json.dumps(data))

Received message for topic: test: {"invoice": 53498, "customerID": 1983, "details": {"discountCode": "1347", "items": [{"sku": 3317, "qty": 460, "description": "action figure : purple : fox"}, {"sku": 5098, "qty": 283, "description": "shoes : beige : kitten"}]}}
Received message for topic: test: {"invoice": 53498, "customerID": 1983, "details": {"discountCode": "1347", "items": [{"sku": 3317, "qty": 460, "description": "action figure : purple : fox"}, {"sku": 5098, "qty": 283, "description": "shoes : beige : kitten"}]}}
```