SwedenDemo Documentation

Release latest

Oct 23, 2019

Contents:

Live Liquidity Forecasting — Using GPS tracking, a distributed ledger and advanced probability analytics, we're creating the next generation liquidity forecasting tool. The goal of this Productive 4.0 demonstrator is to track the whereabouts of **Volvo truck engines** along with their state in the **supply-chain**. This information will be matched with corresponding **truck purchase orders**, in order to generate a live liquidity forecast (FX exposure).

In short: Industry 4.0 meets Bank 4.0.

CHAPTER 1

Project members (in alphabetical order)

- Luleå Technical University
  - Emanuel Palm
  - Olof Schelén
  - Ulf Bodin
- Midroc
  - Oscar Carlsson
- NXP Semiconductors
  - Till Witt
  - Noah Winneberger
- SEB
  - Christian Lagerkvist
  - Caroline Berg von Linde
  - Johan Hörmark
  - Jamie Walters
- Volvo Trucks
  - Richard Hedman

1.1 Introduction

1.1.1 Management summary

Goal of this repository is to provide access to the Swedish usecase demo. The source for this document and the source code can be found at: https://github.com/tlwt/Productive4SwedenDemo

1.1.2 About the Productive 4.0

Productive4.0 is an ambitious holistic innovation project, meant to open the doors to the potentials of Digital Industry and to maintain a leadership position of the industries in Europe. All partners involved will work on creating the capability to efficiently design and integrate hardware and software of Internet of Things (IoT) devices. Linking the real with the digital world takes more than just adding software to the hardware.

1.1.3 Basic architecture

1.2 Creating a Crossbar.io Publisher

1.2.1 Downloading pip and crossbar:

Download pip:

```
$ apt install python3-pip

$ python -m ensurepip
$ python -m pip install -U pip
```

Install crossbar via pip3 (pyhthon3 is required):

```
$ pip3 install crossbar
```

1.2.2 Installing autobahn.js via npm:

Installing nodejs and npm:

```
$ wget https://nodejs.org/dist/v6.10.1/node-v6.10.1-linux-x64.tar.xz
$ tar xvf node-v6.10.1-linux-x64.tar.xz
$ export PATH=${HOME}/node-v6.10.1-linux-x64/bin:${PATH}
$ export NODE_PATH=${HOME}/node-v6.10.1-linux-x64/lib/node_modules
```

Installing autobahn and websocket functionality:

```
$ sudo npm install autobahn
$ sudo npm install ws@2
```

1.2.3 Downloading crossbar libraries and autobahn.js repos:

```
$ git clone https://github.com/crossbario/autobahn-js.git
```

1.3 Creating the GPS Publisher

1.3.1 Configuring the crossbar.io config file:

The important changes to make in regard to the initial default configuration is in the “transports” section, where we define the websocket we want to create (in our case the subdomain autobahn.distributedledger.systems and port 9000):

```
"type": "universal",
"endpoint": {
    "type": "tcp",
    "port": 9000
},
"rawsocket": {},
"websocket": {
    "ws": {
        "type": "websocket",
        "url": "ws://autobahn.distributedledger.systems:9000"
    }
},
```

1.3.2 Initializing the crossbar websocket node:

As seen in the image below (https://crossbar.io/static/img/gen/multi_protocol_on_white_paths.svg) the setup requires the initialization of a crossbar node via which to cennect to in order for a subscriber to successfully receive information published. Building on the changes made to the config.json file starting the crossabr node is simple.

Initalizing the crossbar node:

```
$ crossbar init
$ crossbar start
```

The result should give an output like this:

chapters/../images/crossbar.png

We can now connect to the crossbar node via JavaScript:

```
try {
   // require the autobahn module that was installed via npm, if not globally insert␣
↪the path to the local node_modules folder
   var autobahn = require('autobahn');
} catch (e) {
   // when running in browser, AutobahnJS will
   // be included without a module system
}

var connection = new autobahn.Connection({
   url: 'ws://autobahn.distributedledger.systems:9000/ws',
   realm: 'realm1'}
);
```

For more details and examples how to use the node for publishing and subscribing please refer to the frontend.js and backend.js files

1.4 glossary

todo this still needing correction

fork to be described ;-)

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Index