学位論文

学内の Elasticsearch システムのデータ移行と
冗長化

提出年月日　令和 4 年 X 月 X 日
改定日　　令和 年 月 日

指導教員　　都築　伸二　教授

入学年度　　令和 6 年
学科名　　電子情報工学専攻
論文提出者　　祖父江　匠真

内容梗概

本論文は，筆者が愛媛大学大学院理工学研究科電子情報工学専攻電気電子工学コースに在学中に行った、学内の Elasticsearch のデータ移行と冗長化についてまとめたものであり、以下の 5 章から構成されている。

第 1 章 緒論

本研究を行うに至った経緯及び、本研究の目的について述べている。

第 2 章 学内ゾーンにおける Elasticsearch クラスタへのデータ移行

ここでは学内ゾーンにおける Elasticsearch クラスタへのデータ移行について述べる。

第 3 章 サーバーゾーンでのクラスタ構築における仮想環境を使用した事前検証

ここでは、サーバーゾーンでのクラスタ構築における仮想環境を使用した事前検証について述べている。

第 4 章 サーバーゾーンでのクラスタ構築

ここでは、サーバーゾーンでのクラスタ構築について述べている。

第 5 章 結論

本研究によって明らかになった事項や今後の研究課題について簡単にまとめている。

目 次

第1章

緒論

急速に進化するデータ管理において、Elasticsearch は極めて重要な技術として登場し、データの保存、検索、管理方法に革命をもたらした. 当初、大量のデータを効率的に処理するために設計された Elasticsearch は、シンプルなシングルノードシステムから複雑なクラスタ構成へと移行し、データハンドリング技術において大きな進歩を遂げました (1). この変遷は、様々な分野で堅牢でスケーラブル、かつ冗長性のあるデータ管理システムに対する要求が高まっていることを裏付けている.

データシステムの冗長性、特に Elasticsearch クラスタにおける冗長性は、データの信頼性と可用性を確保する上で重要なポイントとなっています. 冗長性とは、システムの重要なコンポーネントや機能を二重化することで、信頼性を高め、一点障害のリスクを低減することを指す (2). 学内やサーバーゾーンのような複雑な環境では、効果的な冗長性の実現は、特にデータ移行やシステムの拡張性という観点から、独自の課題を提起する. 研究によると、クラスタ化されたシステムにおけるデータ冗長化の取り組みの最大 30%が、初期導入段階で課題に直面しており、このような取り組みの複雑さを浮き彫りにしています (3).

Elasticsearch のノード管理における現在のトレンドは、クラスタ化されたシステムを好む傾向が強まっていることを示しています. しかし、特に学内やサーバゾーンのような分離された環境において、これらのシステムへのデータ移行に関する包括的なドキュメントや分析にはギャップがあります.

Elasticsearch の技術的な側面については多くの研究がありますが、同じ環境内でシングルノードシステムからクラスタ化されたシステムへデータを移行し、さらに別のゾーンに新しいクラスタ化されたシステムを構築する際の実際的な課題や戦略について掘り下げた研究はほとんどありません (4).

本研究の目的は、シングルノードの Elasticsearch システムから学内ゾーン内のクラスタ化システムへデータを移行するプロセスを分析することで、このギャップを埋めることである. 同時に本研究では、この 2 つのゾーン間のデータ移行を行わずに、サーバゾーンに新しい冗長化されたクラスタ化システムを構築することを検討する. この二重のアプローチにより、環境をまたいだデータ移行の複雑さを排除した冗長性と信頼性に焦点を当てた、Elasticsearch データ管理に関するユニークな視点が提供されます.

この研究で期待される成果は、Elasticsearch 環境におけるマイグレーションプロセスの包括的な理解と、別々のゾーンにおける冗長性の確立に関する洞察です. 方法論的に、本研究はケーススタディーアプローチを採用しており、Elasticsearch システムにおけるデータ移行と冗長性の実装の具体例を詳細に分析することができます. このようなアプローチは、より広範な調査や実験的研究 (5) では失われがちな、これらの操作に関わる微妙なプロセスや意思決定を解明する上で特に価値がある.

さらに、この研究はデータ管理のより広い分野に重要な意味を持つ. 得られた洞察は、様々な環境、特に高レベルのデータの信頼性とシステムの冗長性を必要とする環境における将来の Elasticsearch の実装に役立つ可能性がある. さらに、これらの知見は、同様のクラスタ化環境におけるデータ移行プロセスや冗長性戦略の最適化に関する更なる研究に拍車をかける可能性があり、学術研究から産業レベルのデータ管理まで幅広いアプリケーションに恩恵をもたらす可能性がある (6).

要約すると、本研究は Elasticsearch におけるデータ管理の重要な側面である、冗長性を目的としたクラスタシステムへのデータ移行と、異なる環境における別個の、しかし機能的には類似したシステムの確立という、特定の、しかし重要な側面に焦点を当てている点で際立っている. この焦点は、詳細なケーススタディーアプローチと組み合わされ、この分野へのユニークな貢献を提供し、実務家と研究者の両方に貴重な洞察を提供します. 第 2 章では学内

ゾーンにおける Elasticsearch ノードから新クラスタへのデータ移行について述べる．第 3 章では仮想環境を使用してサーバーゾーンの Elasticsearch ノードをシミュレートし、マルチコンテナ Docker アプリケーションのツールである docker-compose を用いてクラスタ構築の実現可能性を検証したことについて述べる．第 4 章ではサーバーゾーンで既存の Elasticsearch ノードを用いたクラスタ構築について述べる．第 5 章では結論と課題を述べる．

第2章

2章のタイトル

2.1節　緒言

本章では … について述べる.

2.2節　•••

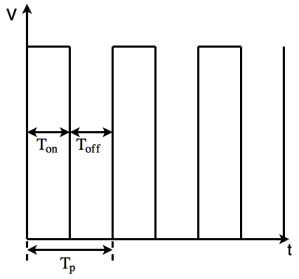

図 2.1　example

2.3節　結言

本章 … について述べた。
次章では … について述べる。

第3章

3章のタイトル

3.1節　緒言

本章では、… ついて述べる。

3.2節　結言

本章では、… について述べた.

第4章

4章のタイトル

4.1節　緒言

本章では…について述べる.

4.2節　結言

本章では…について述べた.

第5章

結論と今後の課題

……

謝 辞

本研究を行うにあたり、終始、懇切丁寧な御指導と適切な御助言を賜りました本学工学部電気電子工学科通信システム工学研究室の都築伸二教授に深甚なる感謝の意を表します。

最後に，有益な御助言を賜りました本大学大学院の〇〇に心より御礼申し上げます。

参考文献

[3]　著者, タイトル, 引用日など.

付録 A

付録

A.1 水源監視システム (送信用 Python スクリプト)

LoRa_obs_transmit.py のソースコードを A.1 に示す.

Listing A.1 LoRa_obs_transmit.py

```
## ++++++coding:utf-8++++++

import time
import os
from datetime import datetime
import serial

""" sleep()をいれて，少し待たないとエラー落ちする """
time.sleep(60)

class Main() :

    def __init__(self):
        """ 初期値および対象ディレクトリの設定 """
        self.s_num = 0

        self.copy_dir = "C:/Users/taikimizukan/Dropbox/sumitomo
          /obscsv/"
        self.target_dir = "C:/Users/taikimizukan/Desktop/
          obs_data/"
```

```
19          self.temporary_log = "./temporary_log.txt"
20
21          """ 最終データを取得 """
22          with open(self.temporary_log,"r") as f :
23              self.old_line = f.readline()
24              print("前回のデータ:"+str(self.old_line))
25
26          """ 起動時 [$RFINF,ONコマンド送信***] """
27          INF = "$RFINF,ON***"
28          with serial.Serial("COM7",115200,timeout=2) as ser :
29              time.sleep(2)
30              while True :
31                  for i in INF :
32                      ser.write(i.encode("utf-8"))
33
34                  result = str(ser.readline())
35                  if result.find("RESULT,RFINF,ON,OK") > 0 :
36                      break
37                  else :
38                      time.sleep(2)
39
40          """ ループ関数実行 """
41          self.Loop()
42
43
44      """ 作成日時が最新ファイルのフルパスを取得し返す関数 """
45      def get_file_path(self, target_dir) :
46          """ 対象ディレクトリ下の.ファイルのパスを取得し
              dat, [target_filesに納める] """
47          target_files = []
48          for root, dir, files in os.walk(target_dir) :
49              target_file = [os.path.join(root,f) for f in files
                  if f.endswith(".dat")]# .txt -> .へ
                  dat
50              target_files.extend(target_file)
51          """ 取得した.ファイルのフルパスに作成時間を足してリストに納める
              dat """
52          file_ctime = []
53          for f in target_files :
54              file_ctime.append((f,os.path.getctime(f)))
55          """ 取得時間でソートし最新の.ファイルのパスのみ返すdat"""
56          sorted_file_ctime = sorted(file_ctime,key=lambda x :x
```

```
            [1])
57
58          return sorted_file_ctime[len(sorted_file_ctime)-1][0]
59
60      """ 最終行を取得，シークエンス番号を加えてコピー  """
61      def check_copy(self):
62  ##          name = self.target_file.replace(self.target_dir,"")
63          with open(self.target_file,"r") as f :
64              """ ファイルデータを全て読み込，最終行だけを取得  """
65              lines = f.readlines()
66              if len(lines) > 0 :
67                  line = lines[len(lines)-1]
68              else :
69                  line = self.old_line
70
71          """ 最終行が前回のものと異なるか？  """
72          if line != self.old_line :
73
74              """ シークエンス番号を追加  """
75              self.s_num += 1
76
77              file_name = line.split(",")
78              file_name = "obs_" +"".join(file_name[0:3])
79              self.ymd = "".join(file_name[0:3])
80
81              self.old_line = line
82
83              """ にコピーDropbox  """
84              with open(self.copy_dir+file_name+".csv","a") as cf
                    :
85                  data = line.strip() + "," + str(self.s_num)+"\n
                      "
86                  cf.write(str(data))
87
88              """ 最終行を保存  """
89              with open(self.temporary_log,"w") as f :
90                  f.write(line)
91
92              self.arduino_serial(data)
93              time.sleep(5)
94              self.TxMSG()
```

```
95              self.Ping()
96
97          else :
98              print("Not␣updated")
99              pass
100
101     """ を経由してにデータを送信する関数  arduinoLoRa"""
102     def arduino_serial(self,d) :
103         print("---"*5 + "arduino_serial" + "---"*5)
104         buf = 0
105         with serial.Serial("COM7",115200,timeout=1) as ser :
106             """ ポートを開いて少し待機が必要 """
107             time.sleep(2)
108             """ごみの吸出し"""
109             buf = ser.readlines()
110             d = d.strip()
111             """ 送信コマンドの形に """
112             d = "$RFSND,0004,"+d+"***"
113             print("To␣arduino␣Data␣-->␣" +d)
114
115             """ Python(PC) -> arduino -> LoRa だと文字ずつ送らないと
                  いけない？1 """
116             for i in d :
117                 ser.write(i.encode("utf-8"))
118
119         """ 0009 : 第二中継機にダミーをとばすMSG, 戻り値を保存 """
120     def TxMSG(self) :
121         target_add = "0009"
122         self.now = datetime.now().strftime("%Y,%m,%d,%H,%M,%S")
123         self.today = datetime.today().strftime("%Y%m%d")
124
125         msg = "$RFSND,{0},{1},{2},{2}***".format(target_add,
              self.now,self.counter)
126
127         with serial.Serial("COM7",115200,timeout=15) as ser :
128                 time.sleep(2)
129                 for i in msg :
130                         ser.write(i.encode("utf-8"))
131                         time.sleep(0.05)
132                 print(ser.readline().decode("utf-8"))
133                 res = ser.readline().decode("utf-8")
```

```
134
135         if len(res) > 10 :
136                 res = res.replace("␣",",").replace("*",",").
                      replace(":",",")
137                 with open("C:/Users/taikimizukan/Dropbox/
                      sumitomo/RSSI_CHECK_TX/rssi_tx_obs_{}.csv".
                      format(str(self.today)),"a") as f :
138                     f.write(res+"\n")
139         else :
140                 pass
141
142     """発電所のにを送って生存確認LoRaping"""
143     def Ping(self) :
144         PING = "$RPING,0004***"
145         with serial.Serial("COM7",115200,timeout=10) as ser :
146             time.sleep(2)
147             for i in PING :
148                 ser.write(i.encode("utf-8"))
149                 time.sleep(0.05)
150
151             print(ser.readline().decode("utf-8"))
152             res_ping = ser.readline().decode("utf-8")
153
154         if len(res_ping) > 10 :
155             print(res_ping)
156             now = datetime.now().strftime("%Y,%m,%d,%H,%M,%S")
157             with open("C:/Users/taikimizukan/Dropbox/sumitomo/
                      PING/ping_{}.csv".format(str(self.today)),"a") as
                       f :
158                     f.write(str(now)+","+str(self.s_num)+"," +
                          str(res_ping))
159         else :
160             pass
161
162     """ 繰り返し """
163     def Loop(self):
164         while True:
165             try :
166                 time.sleep(20)
167                 self.target_file = self.get_file_path(self.
                      target_dir)
```

```
168
169                     self.check_copy()
170
171                 except Exception as E :
172                     with open("./Error_Log.txt","a") as ef :
173                         ef.write(str(E))
174
175
176 if __name__ == "__main__" :
177     Main()
```

A.2 水源監視システム(受信用Pythonスクリプト)

LoRa_obs_receive.py のソースコードを A.2 に示す.

Listing A.2 LoRa_obs_raceive.py

```
1  ## coding:utf-8
2
3  import paho.mqtt.client as mqtt
4  import serial
5  from datetime import datetime
6  import time
7
8  class Main() :
9      def __init__(self) :
10         self.INF_Input()
11         self.Loop()
12
13     def INF_Input(self) :
14     """ 起動時 [$RFINF,ONコマンド送信***] """
15
16         INF = "$RFINF,ON***"
17         with serial.Serial("COM3",115200,timeout=2) as ser :
18             time.sleep(2)
19             while True :
20                 for i in INF :
21                     ser.write(i.encode("utf-8"))
22
23                 result = str(ser.readline())
24                 if result.find("RESULT,RFINF,ON,OK") > 0 :
25                     print("RFINF,OK")
26                     break
27                 else :
28                     time.sleep(2)
29
30     def LoRa_Receive(self) :
31         try :
32         """ からデータを読み込むLoRa """
33             while True :
34                 with serial.Serial("COM3",115200,timeout=120)
                    as ser :
```

```
35                      res = ser.readline().decode("utf-8")
36                      if len(res) > 15 and res.find("RFRX") > 0
                          and res.find("RECEIVED") < 0 :
37                          print("==="*25)
38                          print("RXData␣␣==>␣"+res)
39                          break
40
41                      else :
42                          print("==="*25)
43                          print("else_data=>"+res)
44
45              """ 必要なデータを取り出す """
46              res_list = res.split("*")[0].split(",")[1:]
47              data = ",".join(res_list)
48              add = res_list[0]
49
50              """ データの日付を確認(ymd) """
51              ymd = "".join(res_list[1:4])
52
53
54              return = res, data, add, ymd
55
56          except Exception as E :
57              now = datetime.now().strftime("%y/%m/%d␣%H:%M:%S")
58              with open("LoRa_Receive_Error_LOG.txt","a") as ef :
59                      ef.write(now +"␣:␣"+ str(E)+"\n")
60              self.Loop()
61
62
63
64      def MQTT_Publish(self, res):
65      """ 情報MQTT(publish) """
66          host = "133.71.***.***"
67          port = 1883
68          topic = "***********"
69          """ MQTT-Publish """
70          try :
71              print("publish␣==>␣" +str(res))
72              client = mqtt.Client(protocol=mqtt.MQTTv311)
73              client.connect(host,port=port,keepalive=10)
74              client.publish(topic,res)
```

```
75
76          except Exception as E :
77              now = datetime.now().strftime("%y/%m/%d␣%H:%M:%S")
78              with open("Publish_Error_LOG.txt","a") as ef :
79                      ef.write(now +"␣:␣"+ str(E)+"\n")
80
81      def Storage(self,res,data,ymd):
82          try :
83              """ メタデータを保存 """
84              with open("./meta_data/meta_data_{}.txt".format(ymd
                  ),"a") as f :
85                  f.write(res+"\n")
86
87              """ データを保存 """
88              with open("./data/data_{}.txt".format(ymd),"a") as
                  f :
89                  f.write(data+"\n")
90
91              """ に保存Dropbox """
92              with open("C:/Users/sumitomo02/Dropbox/test_folder/
                  RX/DATA/RX_{}.txt".format(ymd),"a") as f :
93                  f.write(data+"\n")
94
95              with open("C:/Users/sumitomo02/Dropbox/test_folder/
                  RX/META_DATA/RX_{}.txt".format(ymd),"a") as f :
96                  f.write(res+"\n")
97
98          except Exception as E :
99              now = datetime.now().strftime("%y/%m/%d␣%H:%M:%S")
100             with open("Storage_Error_LOG.txt","a") as ef :
101                 ef.write(now +"␣:␣"+ str(E)+"\n")
102             print(E)
103
104     def Loop(self) :
105         while True :
106             res, data, add, ymd = self.LoRa_Receive()
107             self.Storage(res, data, ymd)
108             self.MQTT_Publish(res)
109             self.Ping(add)
110
111
```

```
112 main = Main()
```