3.1.1 Data *Event Log* Private

Data lapangan yang digunakan diperoleh melalui *Terminal Operating Systems* (TOS) yang menyediakan data aktifitas bongkar muat kegiatan impor di PT. Terminal Peti Kemas Surabaya (PT. TPS). Untuk menunjang kebutuhan analisa *process mining* dilakukan pula diskusi terutama kaitannya dalam mengkonfirmasi dan klarifikasi data yang diterima beserta kaitannya dengan proses bisnis di TPS.

Data yang diperoleh dari TPS masih bersifat *flat* karena eksekusi aktifitas disimpan berdasarkan identitas peti kemas yang ditangani. Sedangkan untuk analisa proses mining memperhatikan urutan eksekusi aktifitas berdasarkan waktu. Berikut adalah hasil pengolahan data awal (pra pemrosesan) untuk memperoleh data dalam format standar XES:

a. Data mentah hasil kueri dari basis data *Terminal Operations System* (TOS)

   Data yang diperoleh merupakan hasil kueri dari TOS oleh petugas TPS yang kemudian disimpan dalam format csv. Setiap aktifitas peti kemas dari saat kedatangan masih dalam kapal hingga keluar dari TPS direkam dalam 1 record. Potongan tampilan dari data hasil kueri disajikan pada Gambar 3. 1.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	CONTAINER_KEY	CTR_SIZE	CTR_TYPE	GROSS	VESSEL_ATB	BAPLIE_TS	DISC_DATE	YARD_BLC	YARD_SLC	STACK_DATE	CUSTOMS_DEL_DATE	HAS_QUARANTINE_FLAG	BEHANDLE_QUARANTIN
2	AAMU4001070	40	RFR	23.9	28/08/2021 7:00	26/08/2021 17:32	29/08/2021 1:02	Q	2	29/08/2021 1:17	30/08/2021 0:00		
3	AAMU4001954	40	RFR	33.7	02/08/2021 22:10	02/08/2021 16:46	03/08/2021 0:21	Q	22	03/08/2021 1:22	04/08/2021 0:00	03/08/2021 13:44	
4	AAMU4002523	40	RFR	28.4	28/08/2021 7:00	26/08/2021 17:32	29/08/2021 0:22	ZZ	14	29/08/2021 0:36	29/08/2021 0:00	26/08/2021 17:33	
5	AAMU4002610	40	RFR	23.6	27/08/2021 11:20	27/08/2021 4:32	27/08/2021 13:46	Y	34	27/08/2021 14:08	27/08/2021 0:00	27/08/2021 4:33	
6	AAMU4002713	40	RFR	26.9	02/08/2021 22:10	02/08/2021 16:46	03/08/2021 10:36	YY	14	03/08/2021 11:06	03/08/2021 0:00	02/08/2021 22:27	
7	AAMU4003807	40	RFR	25.7	28/08/2021 20:10	28/08/2021 12:15	28/08/2021 21:31	Z	42	28/08/2021 21:42	29/08/2021 0:00	28/08/2021 12:15	
8	AAMU4004613	40	RFR	33.7	02/08/2021 22:10	02/08/2021 16:46	03/08/2021 5:34	Q	12	03/08/2021 5:58	04/08/2021 0:00	03/08/2021 13:44	
9	AAMU4006616	40	RFR	32.1	27/08/2021 11:20	27/08/2021 4:33	27/08/2021 14:33	B	8	27/08/2021 14:52	27/08/2021 0:00		
10	AAMU4007021	40	RFR	26.5	28/08/2021 20:10	28/08/2021 12:15	28/08/2021 21:58	YY	10	28/08/2021 22:09	30/08/2021 0:00	28/08/2021 12:15	
11	AAMU4007485	40	RFR	33.7	13/08/2021 12:55	13/08/2021 9:34	13/08/2021 15:41	Q	8	13/08/2021 16:18	18/08/2021 0:00	16/08/2021 18:42	
12	AAMU4007546	40	RFR	33.7	13/08/2021 12:55	13/08/2021 9:34	13/08/2021 13:41	B	8	13/08/2021 14:09	16/08/2021 0:00	13/08/2021 9:34	
13	AAMU4007700	40	RFR	33.7	02/08/2021 22:10	02/08/2021 16:46	02/08/2021 23:30	Q	8	03/08/2021 0:57	04/08/2021 0:00	03/08/2021 13:44	
14	AAMU4007926	40	RFR	33.7	02/08/2021 22:10	02/08/2021 16:46	03/08/2021 5:43	Q	12	03/08/2021 6:08	04/08/2021 0:00	03/08/2021 13:44	
15	AAMU4008218	40	RFR	33.7	28/08/2021 7:00	26/08/2021 17:32	28/08/2021 9:38	Q	18	28/08/2021 9:54	30/08/2021 0:00	26/08/2021 17:33	
16	AAMU4008558	40	RFR	28.7	02/08/2021 22:10	02/08/2021 16:46	03/08/2021 5:45	Q	12	03/08/2021 6:10	07/08/2021 0:00	05/08/2021 17:27	
17	AAMU4009528	40	RFR	25.9	02/08/2021 22:10	02/08/2021 16:46	03/08/2021 0:18	Q	22	03/08/2021 1:28	05/08/2021 0:00	04/08/2021 11:17	
18	AKLU6006170	20	DRY	3.319	21/08/2021 20:25	20/08/2021 22:15	21/08/2021 21:31	L	105	21/08/2021 21:45	23/08/2021 0:00		
19	AKLU6018802	20	DRY	22.8	10/08/2021 20:15	10/08/2021 9:44	11/08/2021 8:54	L	99	11/08/2021 9:26	13/08/2021 0:00		
20	AKLU6019075	20	DRY	22.21	21/08/2021 20:25	20/08/2021 22:17	22/08/2021 2:15	O	83	22/08/2021 2:42	23/08/2021 0:00		
21	AKLU6026480	20	DRY	29.55	31/07/2021 14:55	30/07/2021 14:03	31/07/2021 19:30	M	37	31/07/2021 19:46	03/08/2021 0:00		
22	AKLU6029643	20	DRY	30.25	21/08/2021 20:25	20/08/2021 22:15	22/08/2021 0:38	K	131	22/08/2021 0:55	23/08/2021 0:00		

tps_Agustus2021

Gambar 3. 1 Potongan data log dari Terminal Operations System (TOS)

Terdapat beberapa ketentuan yang perlu diperhatikan untuk mengubah data mentah pada Gambar 3. 1 agar sesuai standar XES:

1. Tiap *record* adalah data event (dari suatu case)
2. Tiap event terkumpul secara urut waktu dalam satu case
3. Tiap case hanya berisi *event* (aktifitas) yang aktif
4. Tiap case harus memiliki ID unik

5. Perlu membedakan antara data dengan cakupan case dan data yang hanya dihasilkan oleh *event* tertentu

b. Menggabungkan data per bulan dan mengubah ke Dataframe Pandas

Penggabungan data tiap bulan dan mengubahnya menjadi Dataframe Pandas

```
df_07 = pd.read_excel('./data/tps_Juli2021.xls', delimiter=";")
df_08 = pd.read_excel('./data/tps_Agustus2021.xls', delimiter=";")
df_09 = pd.read_excel('./data/tps_September2021.xls', delimiter=";")
```

```
# vertical concat
df_all = pd.concat([df_07, df_08, df_09], axis=0)
```

Informasi data hasil penggabungan meliputi nama atribut dan tipe datanya.

```
df_all.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 95648 entries, 0 to 32338
Data columns (total 28 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   CONTAINER_KEY                   95648 non-null  object
 1   CTR_SIZE                        95648 non-null  int64
 2   CTR_TYPE                        95648 non-null  object
 3   GROSS                           95648 non-null  float64
 4   VESSEL_ATB                      95648 non-null  datetime64[ns]
 5   BAPLIE_TS                       95648 non-null  datetime64[ns]
 6   DISC_DATE                       95648 non-null  datetime64[ns]
 7   YARD_BLOCK                      95624 non-null  object
 8   YARD_SLOT                       95406 non-null  float64
 9   STACK_DATE                      95624 non-null  datetime64[ns]
 10  CUSTOMS_DEL_DATE                95648 non-null  datetime64[ns]
 11  HAS_QUARANTINE_FLAG             26940 non-null  datetime64[ns]
 12  BEHANDLE_QUARANTINE_JOB_TS      18 non-null     datetime64[ns]
 13  FIRST_STACK_QUARANTINE_BLK_TS   13639 non-null  datetime64[ns]
 14  QUARANTINE_RELEASE              26940 non-null  datetime64[ns]
 15  JOB_PLP_TS                      505 non-null    datetime64[ns]
 16  JOB_DEL_DATE                    95147 non-null  datetime64[ns]
 17  JOB_DEL_DOCTYPE                 95647 non-null  object
 18  TRUCK_IN_DATE                   95647 non-null  datetime64[ns]
 19  UNSTACK_TO_TRUCK                95540 non-null  datetime64[ns]
 20  TRUCK_OUT_DATE                  95647 non-null  datetime64[ns]
 21  SP_JALUR_MERAH_PIB_DATE         5155 non-null   datetime64[ns]
 22  CUSTOMS_BEHANDLE_COUNT          95648 non-null  int64
 23  FIRST_JOB_CUSTOMS_BEHANDLE_TS   3924 non-null   datetime64[ns]
 24  FIRST_STACK_BEHANDLE_TS         4032 non-null   datetime64[ns]
 25  LAST_COMPLETE_BEHANDLE_TS       3924 non-null   datetime64[ns]
 26  STACK_TRK_BF_MAIN_YARD_TS       2951 non-null   datetime64[ns]
 27  STACK_MAIN_YARD_TS              3844 non-null   datetime64[ns]
dtypes: datetime64[ns](20), float64(2), int64(2), object(4)
memory usage: 19.7+ MB
```

c. Transformasi ke struktur *event log* mengacu format XES

Langkah-langkah:

1. Menentukan case ID

   CONTAINER_KEY tidak dapat digunakan sebagai case ID karena peti kemas yang sama dapat datang pada waktu yang berbeda sehingga tidak unik. Untuk menjadikannya unik maka perlu menggabungkan CONTAINER_KEY dengan tanggal kedatangan.

```
df_seleksi.head()
```

	ID	CONTAINER_KEY	CTR_SIZE	CTR_TYPE	GROSS	YARD_BLOCK	YARD_SLOT	JOB_DEL_DOCTYPE	VESSEL_ATB	BAPLIE	...	JOB_DEL	T
0	AAAU9001220-2021-06-28	AAAU9001220	40	DRY	12.20	N	98.0	SPPB	2021-06-28 20:00:00	2021-06-27 11:09:00	...	2021-07-05 17:13:00	
1	AAMU4003031-2021-06-20	AAMU4003031	40	RFR	31.30	Q	8.0	SPPB	2021-06-20 12:45:00	2021-06-19 14:50:00	...	2021-06-29 15:56:00	
2	AAMU4004969-2021-07-07	AAMU4004969	40	RFR	33.70	B	8.0	SPPB	2021-07-07 15:50:00	2021-07-06 19:09:00	...	2021-07-08 21:34:00	
3	AAMU4008291-2021-07-07	AAMU4008291	40	RFR	33.70	B	18.0	SPPB	2021-07-07 15:50:00	2021-07-06 19:09:00	...	2021-07-08 21:34:00	
4	ACCU2049701-2021-07-13	ACCU2049701	20	DRY	30.08	J	111.0	SPPB	2021-07-13 20:20:00	2021-07-13 16:51:00	...	2021-07-21 09:53:00	

2. Mengenali dan membedakan antara atribut yang merupakan aktifitas dengan atribut sebagai data.

```
df_data = df_seleksi.iloc[:,1:8]
df_data.head()
```

	CONTAINER_KEY	CTR_SIZE	CTR_TYPE	GROSS	YARD_BLOCK	YARD_SLOT	JOB_DEL_DOCTYPE
0	AAAU9001220	40	DRY	12.20	N	98.0	SPPB
1	AAMU4003031	40	RFR	31.30	Q	8.0	SPPB
2	AAMU4004969	40	RFR	33.70	B	8.0	SPPB
3	AAMU4008291	40	RFR	33.70	B	18.0	SPPB
4	ACCU2049701	20	DRY	30.08	J	111.0	SPPB

Menampung tiap data hanya pada aktifitas (*event*) yang sesuai:

```
df_data_baplie = df_data.copy()
df_data_baplie.iloc[:,3:7] = None
df_data_baplie.head() # data baplie adl CTR_SIZE DAN CTR_TYPE
```

	CONTAINER_KEY	CTR_SIZE	CTR_TYPE	GROSS	YARD_BLOCK	YARD_SLOT	JOB_DEL_DOCTYPE
0	AAAU9001220	40	DRY	None	None	None	None
1	AAMU4003031	40	RFR	None	None	None	None
2	AAMU4004969	40	RFR	None	None	None	None
3	AAMU4008291	40	RFR	None	None	None	None
4	ACCU2049701	20	DRY	None	None	None	None

```
df_data_discharge = df_data.copy()
df_data_discharge.iloc[:,1:3] = None
df_data_discharge.iloc[:,6:7] = None
df_data_discharge.head() # DATA DISCHARGE ADL GROSS, YARD BLOCK, DAN YARD SLOT
```

	CONTAINER_KEY	CTR_SIZE	CTR_TYPE	GROSS	YARD_BLOCK	YARD_SLOT	JOB_DEL_DOCTYPE
0	AAAU9001220	None	None	12.20	N	98.0	None
1	AAMU4003031	None	None	31.30	Q	8.0	None
2	AAMU4004969	None	None	33.70	B	8.0	None
3	AAMU4008291	None	None	33.70	B	18.0	None
4	ACCU2049701	None	None	30.08	J	111.0	None

```
df_job_del = df_data.copy()
df_job_del.iloc[:,1:6] = None
df_job_del.head() #
```

	CONTAINER_KEY	CTR_SIZE	CTR_TYPE	GROSS	YARD_BLOCK	YARD_SLOT	JOB_DEL_DOCTYPE
0	AAAU9001220	None	None	None	None	None	SPPB
1	AAMU4003031	None	None	None	None	None	SPPB
2	AAMU4004969	None	None	None	None	None	SPPB
3	AAMU4008291	None	None	None	None	None	SPPB
4	ACCU2049701	None	None	None	None	None	SPPB

Mengidentifikasi atribut aktifitas dan menyeleksi aktifitas yang aktif

```
df_activities = df_seleksi.iloc[:,8:] # kolom aktifitas
df_activities.head()
```

	VESSEL_ATB	BAPLIE	DISCHARGE	STACK	CUSTOMS_DEL	HAS_QUARANTINE_FLAG	BEHANDLE_QUARANTINE_JOB	FIRST_STACK_QUARANTINE_BLK
0	2021-06-28 20:00:00	2021-06-27 11:09:00	2021-06-28 22:22:00	2021-06-28 22:40:00	2021-07-05	2021-07-02 09:06:00	NaT	2021-07-03 01:43:00
1	2021-06-20 12:45:00	2021-06-19 14:50:00	2021-06-20 14:34:00	2021-06-20 14:51:00	2021-06-25	NaT	NaT	NaT
2	2021-07-07 15:50:00	2021-07-06 19:09:00	2021-07-07 17:32:00	2021-07-07 18:21:00	2021-07-08	2021-07-07 10:50:00	NaT	NaT
3	2021-07-07 15:50:00	2021-07-06 19:09:00	2021-07-08 11:46:00	2021-07-08 12:31:00	2021-07-08	2021-07-07 10:50:00	NaT	NaT
4	2021-07-13 20:20:00	2021-07-13 16:51:00	2021-07-13 23:25:00	2021-07-14 00:16:00	2021-07-19	NaT	NaT	NaT

```
# MEnyeleksi aktifitas yang memiliki timestampn (aktifitas tanpa timestamp artinya tidak diaktivasi)
col = column # id and activities
eventLog = []
for i in range(len(df_activities)):
    case = df_activities.iloc[i] # telusuri case i (berisi timestamp)
    case_fix = []
    for j in range(len(case)):
        ts = case[j] # timestamp tiap activity
        if isinstance(ts, pd.Timestamp): # hanya ambil yang ada timestamp nya
            case_fix.append({'activity': col[j+8], 'timestamp': ts}) # +8 karena skip kolom id dan data
    eventLog.append(case_fix)
# eventLog
```

```
eventLog[3]
```

```
[{'activity': 'VESSEL_ATB', 'timestamp': Timestamp('2021-07-07 15:50:00')},
 {'activity': 'BAPLIE', 'timestamp': Timestamp('2021-07-06 19:09:00')},
 {'activity': 'DISCHARGE', 'timestamp': Timestamp('2021-07-08 11:46:00')},
 {'activity': 'STACK', 'timestamp': Timestamp('2021-07-08 12:31:00')},
 {'activity': 'CUSTOMS_DEL', 'timestamp': Timestamp('2021-07-08 00:00:00')},
 {'activity': 'HAS_QUARANTINE_FLAG',
  'timestamp': Timestamp('2021-07-07 10:50:00')},
 {'activity': 'QUARANTINE_RELEASE',
  'timestamp': Timestamp('2021-07-08 21:22:00')},
 {'activity': 'JOB_DEL', 'timestamp': Timestamp('2021-07-08 21:34:00')},
 {'activity': 'TRUCK_IN', 'timestamp': Timestamp('2021-07-08 23:13:00')},
 {'activity': 'UNSTACK_TO_TRUCK',
  'timestamp': Timestamp('2021-07-08 23:22:00')},
 {'activity': 'TRUCK_OUT', 'timestamp': Timestamp('2021-07-08 23:28:00')}]
```

Event log dalam tiap case harus diurutkan berdasarkan timestamp

```
# sort berdasarkan timestamp

def myFunc(e):
    return e['timestamp']
eventLog_fix = []
for case in eventLog:
    case.sort(key=myFunc) # urutkan aktifitas2 dalam case
    case_fix = []
    for event in case:
        case_fix.append([event['activity'], event['timestamp']])
    eventLog_fix.append(case_fix)
```

```
eventLog_fix[2] # case ke-x berisi event-event utk case ke-x
```

```
[['BAPLIE', Timestamp('2021-07-06 19:09:00')],
 ['HAS_QUARANTINE_FLAG', Timestamp('2021-07-07 10:50:00')],
 ['VESSEL_ATB', Timestamp('2021-07-07 15:50:00')],
 ['DISCHARGE', Timestamp('2021-07-07 17:32:00')],
 ['STACK', Timestamp('2021-07-07 18:21:00')],
 ['CUSTOMS_DEL', Timestamp('2021-07-08 00:00:00')],
 ['QUARANTINE_RELEASE', Timestamp('2021-07-08 21:22:00')],
 ['JOB_DEL', Timestamp('2021-07-08 21:34:00')],
 ['TRUCK_IN', Timestamp('2021-07-08 23:21:00')],
 ['UNSTACK_TO_TRUCK', Timestamp('2021-07-08 23:39:00')],
 ['TRUCK_OUT', Timestamp('2021-07-08 23:45:00')]]
```

Langkah akhir adalah menyematkan data ke aktifitas (event) yang sesuai.

```
# Langkah akhir menyusun Event Log mengacu format XES
col = column
case_id_fix = [item for sublist in df_case_id.values.tolist() for item in sublist] # case id
# data_fix = [sublist for sublist in df_data.values.tolist()] # tiap data dalam list
data_baplie = [sublist for sublist in df_data_baplie.values.tolist()] # tiap data dalam list
data_discharge = [sublist for sublist in df_data_discharge.values.tolist()] # tiap data dalam list
data_job_del = [sublist for sublist in df_job_del.values.tolist()] # tiap data dalam list
data_other = [sublist for sublist in df_data_other.values.tolist()] # tiap data dalam list

eventLog_fix = eventLog_fix

eventLog_final = []
for i in range(len(eventLog_fix)): # jumlah case, untuk tiap case
    case_final = []
    for events in eventLog_fix[i]: # ['BAPLIE_TS', Timestamp('2021-04-27 15:30:00')] --> case ke-i
#         event = []
        event = [case_id_fix[i]] # +case id
#         event.extend(data_fix[i]) # +kolom2 data level case
        if events[0] == 'BAPLIE':
            event.extend(data_baplie[i]) # tambahkan data pada baplie
        elif events[0] == 'DISCHARGE':
            event.extend(data_discharge[i]) # tambahkan data pada discharge
        elif events[0] == 'JOB_DEL':
            event.extend(data_job_del[i]) # tambahkan data pada job_del
        else:
            event.extend(data_other[i]) # aktivitas tanpa data kosong
        for e in events: # +activity dan timestamp ['BAPLIE_TS', Timestamp('2021-04-27 15:30:00')] --> tiap event dalam case ke-i
            event.append(e)
        case_final.append(event) # tambahkan event ke case ['CAIU3407429-2021-04-27 07:10:00','CAIU3407429',20,'DRY',3.03,'N',61.0
    eventLog_final.append(case_final)
```

Menambahkan nama kolom sehingga diperoleh hasil akhir yang lengkap.

```
data = [activity for event in eventLog_final for activity in event]
df_eventlog_final = pd.DataFrame(data)
df_eventlog_final.columns = ['case_id', 'container key','CTR_SIZE','CTR_TYPE','GROSS','YARD_BLOCK','YARD_SLOT','JOB_DEL_DOCTYPE','
```

```
df_eventlog_final[:20]
```

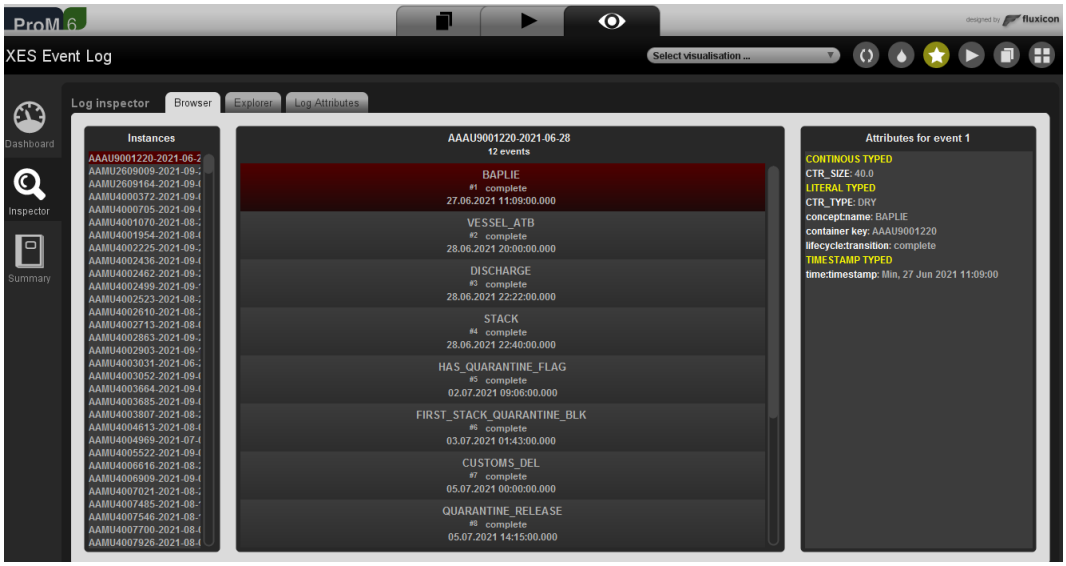
	case_id	container key	CTR_SIZE	CTR_TYPE	GROSS	YARD_BLOCK	YARD_SLOT	JOB_DEL_DOCTYPE	activity	timestam
0	AAAU9001220-2021-06-28	AAAU9001220	40.0	DRY	NaN	None	NaN	None	BAPLIE	2021-0 2 11:09:0
1	AAAU9001220-2021-06-28	AAAU9001220	NaN	None	NaN	None	NaN	None	VESSEL_ATB	2021-0 2 20:00:0
2	AAAU9001220-2021-06-28	AAAU9001220	NaN	None	12.2	N	98.0	None	DISCHARGE	2021-0 2 22:22:0
3	AAAU9001220-2021-06-28	AAAU9001220	NaN	None	NaN	None	NaN	None	STACK	2021-0 2 22:40:0
4	AAAU9001220-2021-06-28	AAAU9001220	NaN	None	NaN	None	NaN	None	HAS_QUARANTINE_FLAG	2021-0 0 09:06:0
5	AAAU9001220-2021-06-28	AAAU9001220	NaN	None	NaN	None	NaN	None	FIRST_STACK_QUARANTINE_BLK	2021-0 0 01:43:0
6	AAAU9001220-2021-06-28	AAAU9001220	NaN	None	NaN	None	NaN	None	CUSTOMS_DEL	2021-0 0 00:00:0

Export data event log ke format csv.

3. Mengubah data ke format XES

   Untuk mengubah format csv ke XES dapat menggunakan plugin pada ProM.

Hasil konversi ke XES :



Hasil ini menunjukkan bahwa semua data dapat terbaca dengan benar sehingga data event log ini dapat digunakan untuk eksperimen *process mining* selanjutnya.