**PROCESS MINING ASSIGNMENT**

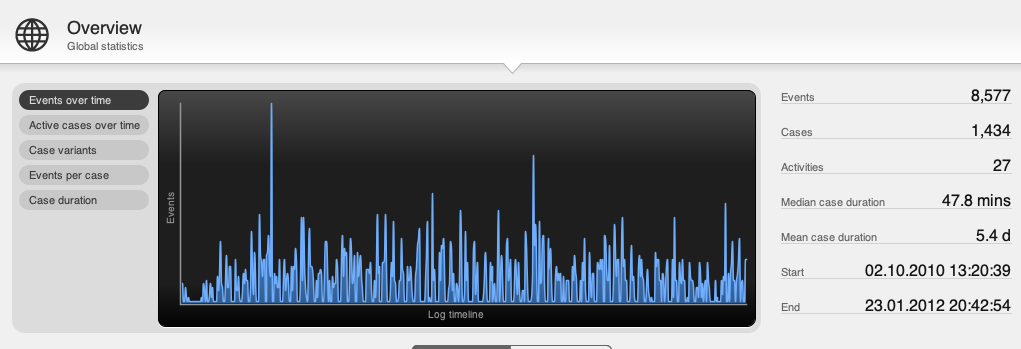
Applying Process Mining on Real Data

**QUESTION 1**

Open the event log ('Receipt phase of an environmental permit application process (\_WABO\_) CoSeLoG project.fbt') in Disco and switch to the 'Statistics' view.

Without switching to other views, use the statistics view to answer the following three sub questions:

1. How many events are there on average per case?

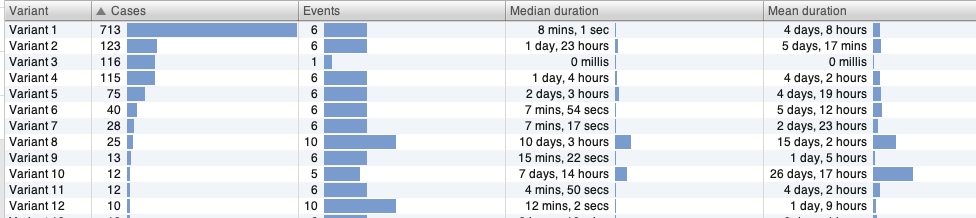


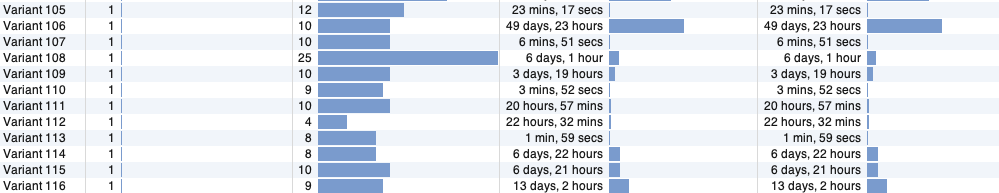
events = 8577

cases = 1434

therefore 8577/1434 = 5.98 events/case on avg

1. Can you indicate whether each case seems to be unique or whether many cases follow the same activity sequence?

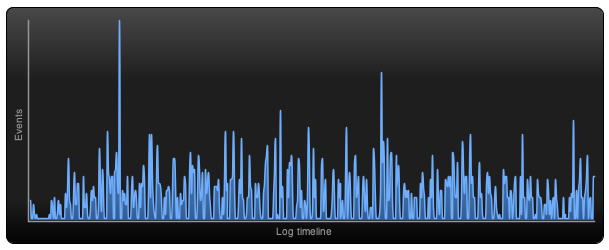




It seems that many cases are not unique. The top 4 variants for instance cover 1000 of the 1434 cases and that most cases follow one of few variants, but there is a 'tail' of cases that are all unique.

About 50% of all cases follow one activity sequence, there are in total 116 sequence variants, with a total of 1434 cases. Most cases are *not* unique, but variants 31 to 116 are single case variants, so these cases follow a unique activity sequence.

1. What is the main observation that can be made from the 'Events over time' graph?



We can see that the amount of activities per day is in a range of 0 to 147. There are many groups of 2 days or more with zero activities. Probably weekends and holidays.

**QUESTION 2**

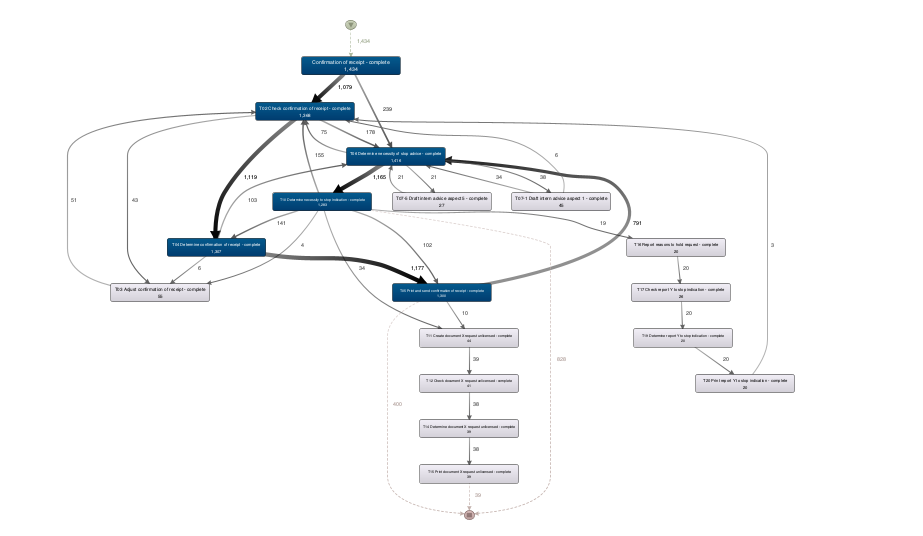
While still in Disco, switch to the 'map' view to display a process map.

Using the map view, change the activity and path detail settings in order to create a comprehensible process map (e.g. a process map that could be printed on one A4 or letter paper or shown on a single computer screen while still being readable in full). In your answer, include the settings you used for both the activity and path sliders.

1. Discuss this process map, what is the main process?

Switching to the process map for 100 percent of activities and 0 percent of path is not comprehensible, it contains too many different activities. On the other hand only a minor number of traces finish at the end event. So the number of activities has to be decreased and the number of paths increased.

The best compromise seems to be 50% of activities and 17% of paths. Now nearly 90% of traces finish at the end event and the most important activities are visible.



The major parts of the main process are ‘Confirmation of receipt’ and ‘Determine necessity of stop’. ‘Confirmation of receipt’ is self-declaring, ‘necessity of stop’ not at all, so it is difficult to comprehend what might be stopped.

1. Which activities and paths between activities are frequent?

The main process consists of 2 groups of most frequent activities, conducted a few times parallel but most times in sequence:

* Confirmation of receipt
* Check confirmation of receipt
* Determine confirmation of receipt
* Print and sent confirmation of receipt
* Determine necessity of stop advice
* Determine necessity of stop indication

About 90 % of all cases contain these 6 activities.

**QUESTION 3**

While still in Disco, and while using the same process map (e.g. do not change the activity and path settings), switch to the performance projection.

Discuss where the process takes most time, e.g. where there are possibilities for improvement. Relate these times (of the bottlenecks) to the time spent in other parts of the process. In other words, discuss how severe the bottleneck is with respect to the time spent on other activities.

Also, explicitly mention the performance metric chosen (e.g. total, mean, median, or max) and why you have chosen this setting.

**ANALYSIS**

* Switching to the performance projection.
* Keeping the sliders at 50.8% activities and 17.4% path
* Setting Show to ‘Mean duration’
* Adding as Secondary ‘Absolute frequencies’

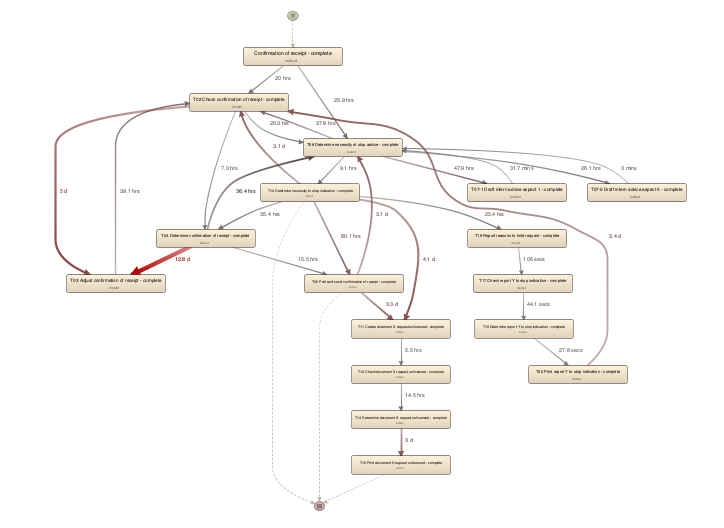
Most time spent affecting a majority of cases is the “T06 determine necessity to stop advice activity” as can be seen in the graph, 3.1days in 791 events. If this could be shortened significantly, the response times would improve much and stake holders would have to wait much less.  
There are other even longer lasting events, but those are happening a lot less often so they are not significant as much as the above mentioned.

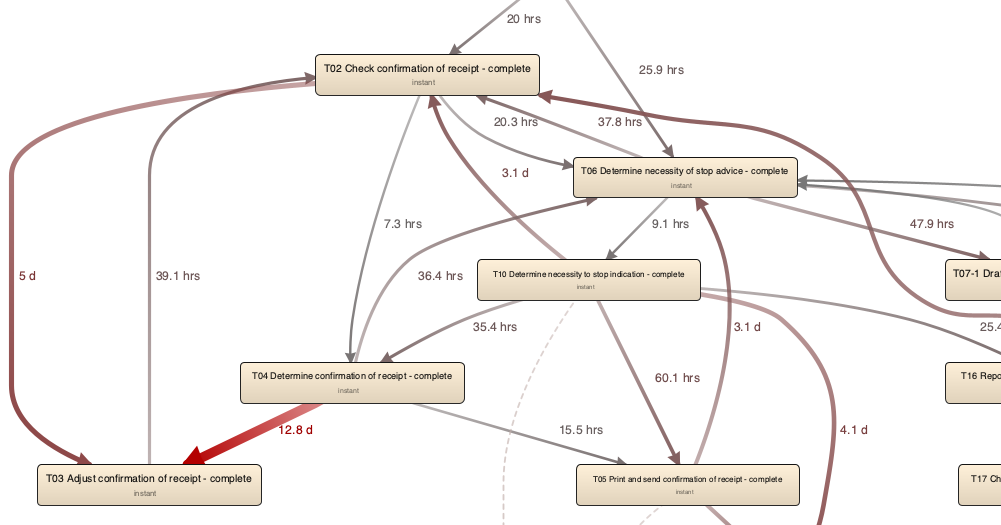
Generally, there are 2 important perspectives when analyzing process durations.

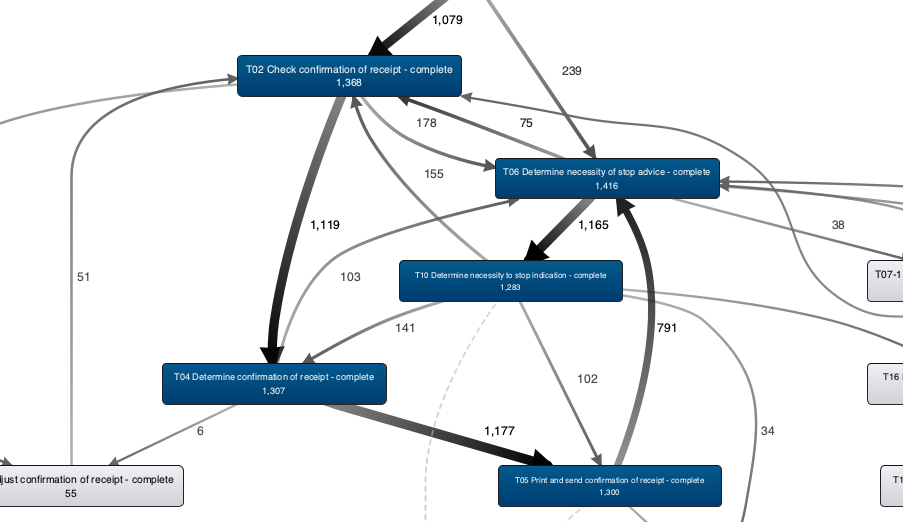
* Total durations of an activity for all cases is the relevant indicator if you want to find out where most effort went into, which typically relates which activities are the most expensive. Depending on whether these activities are evenly distributed over many parallel workers or not, the total duration significantly indicates extraordinary long waiting times as well.
* Mean duration of activities and cases (full traces) is also relevant if you want to find out how long external stake holders of a case have to wait until it is finished.

The focus here is on the external perspective - what are major factors for long case run times?

In this process mining case study, total time and long mean run time are found at the same activity (79.6 mth, 791 activities). So addressing this is an improvement process change that would be beneficial for both external stake holders and costs.







**QUESTION 4**

Now load the original event log in ProM. Visualize the event log using the Dotted Chart or XDottedChart visualizer (by pressing the 'eye'-icon with the event log selected and switching to the Dotted Chart or XDottedChart visualizer).

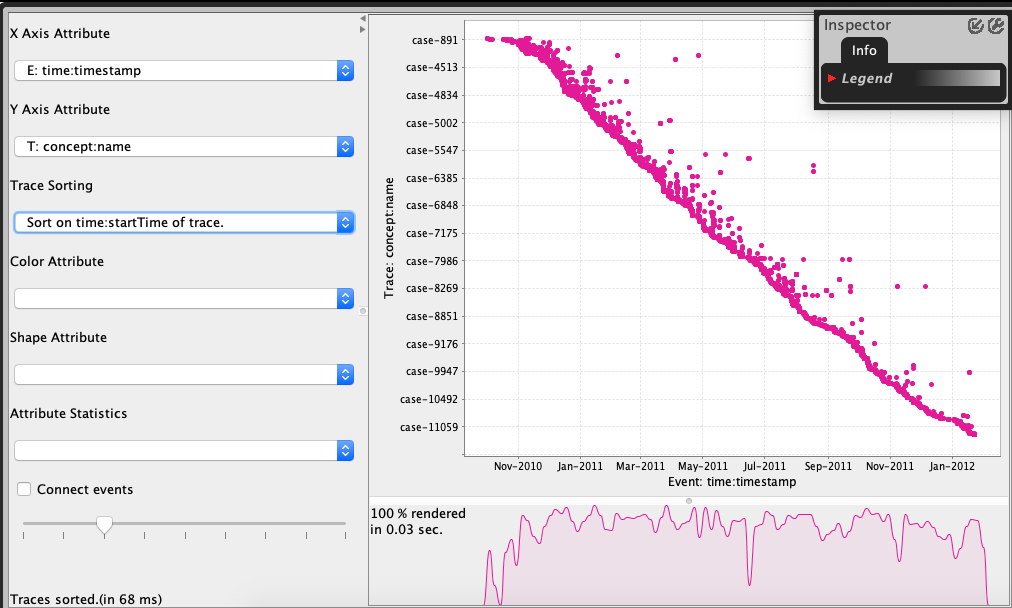
Using the Dotted Chart, answer the following questions:

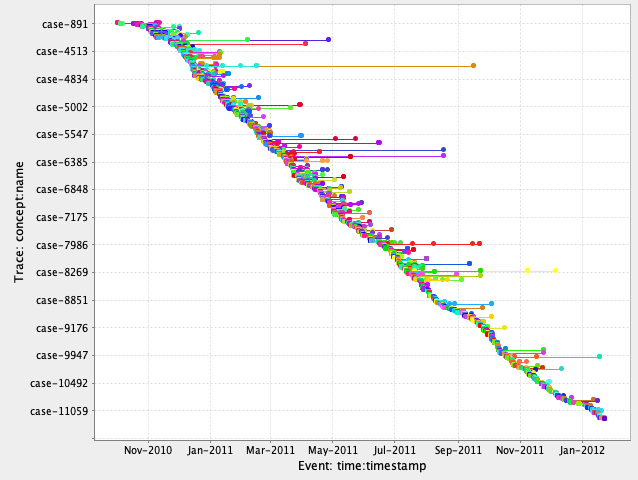
1. Is the arrival rate of new cases constant? If not, when are there fluctuations? If yes, how can we see this from the Dotted Chart?
2. Can you observe a change in the global process?

Note that you don't need to change the component, time or coloring settings. You can however re-sort the traces on the time of the first event, and zoom in or out if you want.

**SOLUTION**

While sorting by time: startTime of Trace gives an overview of case start rate. It is magnified here:





From the diagram above, it can be concluded:

* There are small fluctuations in the arrival rate.
* There is a general trend for a slightly lower rate of new cases from the month of September.

**QUESTION 5**

You are now asked to discover a Petri net on the event log. However, the unfiltered event log results in an incomprehensible Petri net. Therefore, you are allowed to run the 'Filter log using simple heuristics' plug-in *once* on the original event log to discover a Petri net on the filtered event log.

1. Clearly indicate which settings you have used for the 'Filter log using simple heuristics' plug-in.
2. Explicitly motivate the filtering settings chosen, why did you pick this percentage or selection of activities?
3. Discuss and argue which plug-in (or chain of plug-ins) you have used to discover a Petri net, for instance by comparing two or more plug-in results and arguing why one of the Petri nets is better.
4. Explain the (best) Petri net: what is the main process and what are notable parts of the Petri net?

Note that this question requires you to experiment with different filtering settings and discovery plug-ins. You are not required to describe *everything* you have tried but found unsuccessful. Only describe the successful combination of plug-ins and its result(s) and argue why your final result is 'good'.

**Suggested list of plug-ins or plug-in chains to produce a Petri net:**

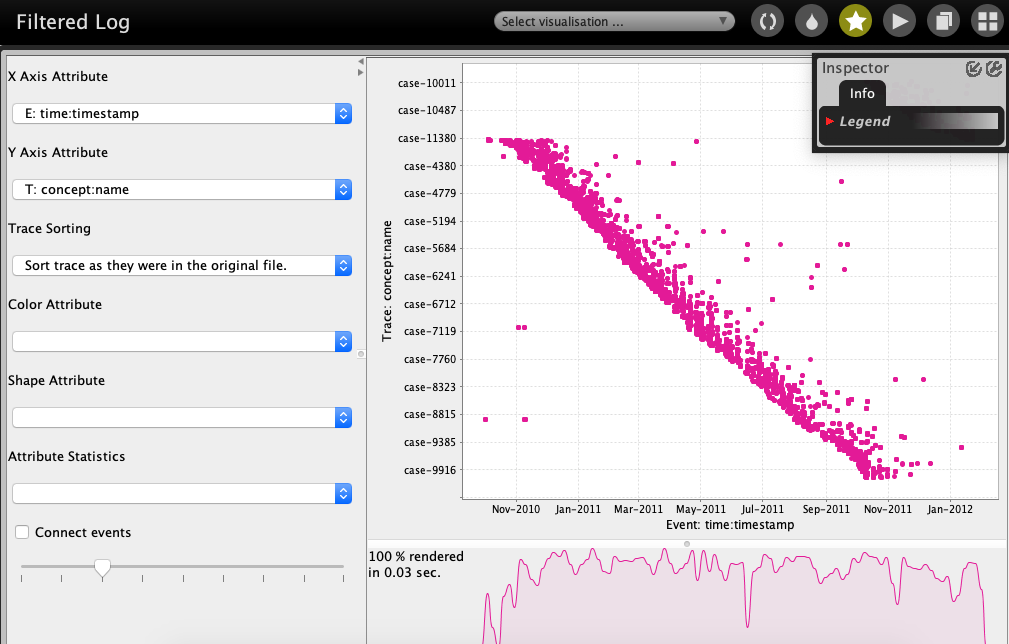
* Mine for a Petri Net using Alpha-algorithm
* Mine for a Petri Net using ILP
* Mine for a Heuristics Net using Heuristics Miner *followed by* Convert Heuristics net into Petri net
* Mine for a Petri net with Inductive Miner

**ANALYSIS**

**Settings used for the 'Filter log using simple heuristics' plug-in.**

* 1. Click on “Actions” icon.
  2. Search for “Filter Log”.
  3. Select “Filter Log using Simple Heuristics”.
  4. Click on “Start” button.
  5. Change Log name to “Filtered Log” .
  6. Click on “Next” button.
  7. Select “Select top percentage” to 100% because there is only 1 Start event.
  8. Click on “Next” button.
  9. Select “Select top percentage” to 100% because ideally keeping all End events would be critical in understanding the process.
  10. Click on “Next” button.
  11. Select “Select top percentage” to 96% because this Event filter criterion discards many events and therefore many arcs in the resulting Petri net.
  12. Click on “Finish” button.

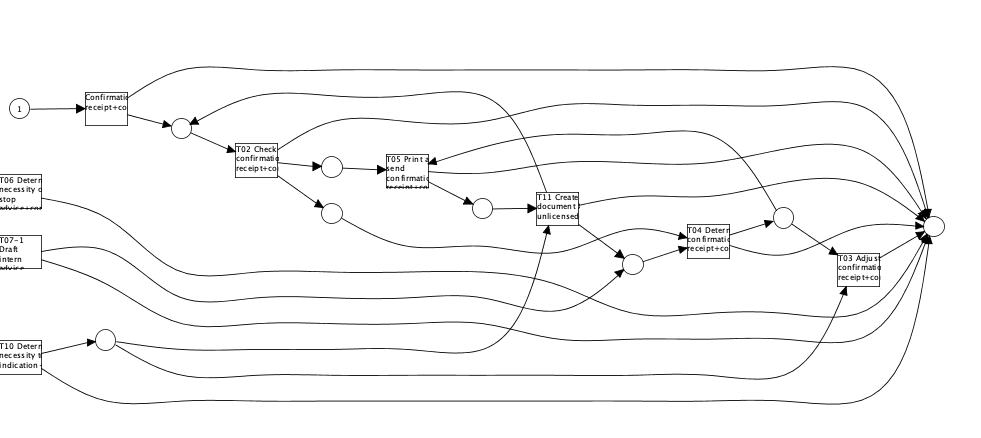




**Mine for a Petri Net using Alpha-algorithm**

1. Click on “Actions” icon.  
2. Add filtered log to “Input”.  
3. Select “Mine for a Petri Net using Alpha-algorithm” plug in.

4. Click on “Start” button.

****

The Alpha algorithm has discovered 9 transitions & 9 places. However, transitions T06, T07-1 & T10 are not integrated well into the rest of the control-flow.

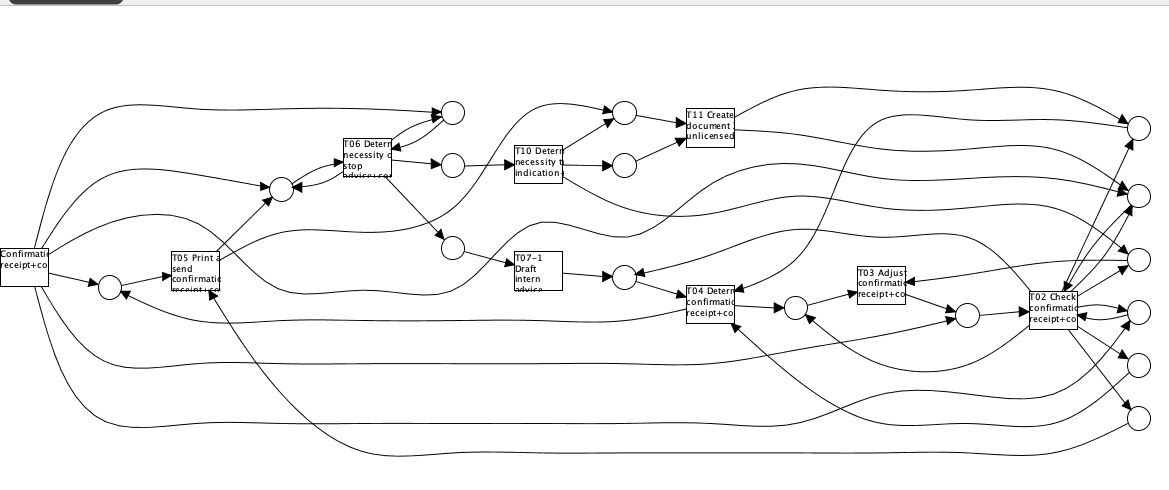
However “Mine for a Petri Net using Alpha-algorithm” plug-in is not robust to logs that contain noisy data (like real-life logs typically do).

The mining plug-ins “Flexible Heuristics Miner”, “Inductive Miner”, and “Fuzzy Miner” is good to go for real-life logs.

**Mine for a petri net using ILP**

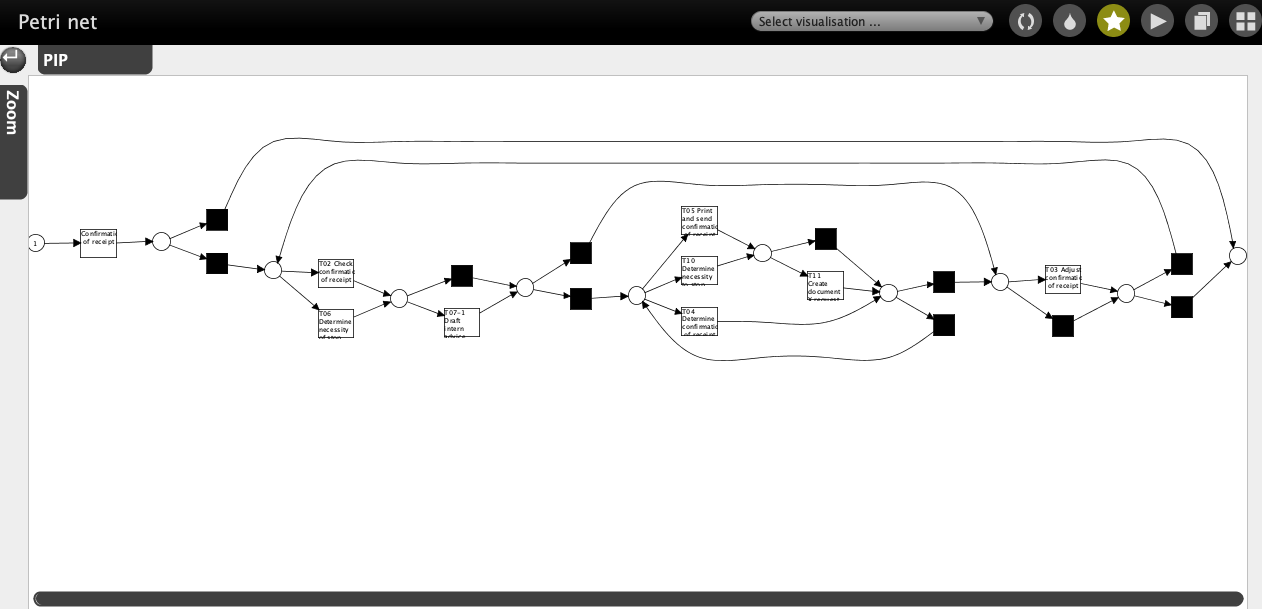
1. Click on “Actions” icon.
2. Add “CoSeLoG (96% filtered. . . )” log to “Input”.
3. Search for “ILP” plug-in.
4. Select “Mine for a Petri Net using ILP”.
5. Click on “Start” button.
6. Select the “Number of places” option to “Before & After Transition” instead of “Per Causal Dependency” to ensure clear “End” states & minimize number of arcs.

(select Detailed configuration: deselect empty after completion, LP Objective: minimize arcs, Discovery strategy: mine a place per causal relation)

1. Click “Finish” button.   
   

**Mine for a petri net using Inductive Miner**

1. Click on “Workspace” icon.
2. Select filtered\_event log.
3. Click on “Actions” icon.
4. Search for “Inductive” plug-in.
5. Select “Mine Petri net with Inductive Miner” plug-in.
6. Click on “Start” button.
7. Change “Variant” option from default of “Inductive Miner - infrequent” to “Inductive Miner” because the default option drops T04 transition probably due to infrequent cases containing it. We want to keep this transition so that we can compare the different Petri nets with the same set of transitions.
8. Click “Finish” button.



Nearly as good results can be achieved with other miners, but none gave a better net and none could be used to find the optimal filter settings for the best compromise of detail and drop of distorting traces.

The essential step to discover a comprehensible Petri net from the supplied log data is to filter out classes of traces with only a few class members but which are complicating the process model without giving extra insights. The major challenge is setting the filters to such a level that an optimum of the real process variants is captured. Whether a class of traces is included or not for a good process model should not depend on miner chosen for a Petri net. But there are differences in features a process discovery method is able to guarantee.

**QUESTION 6**

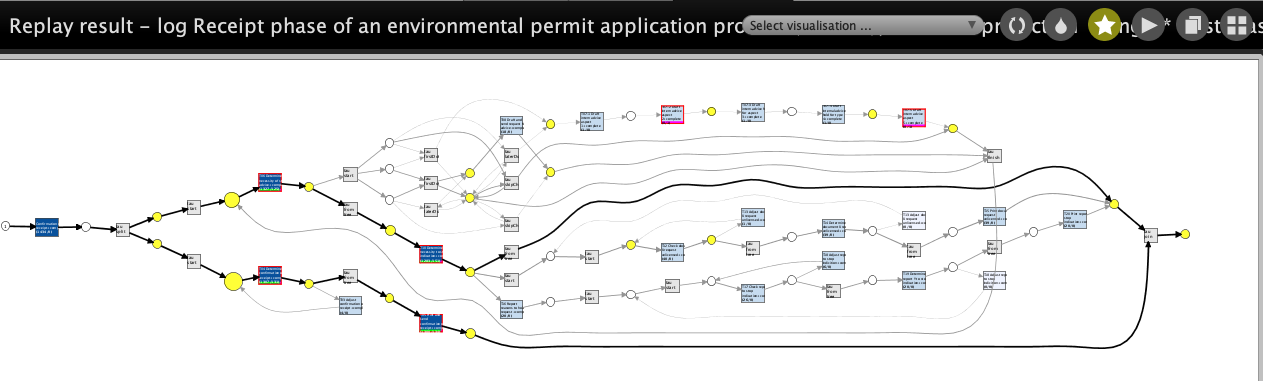
The organization has a process model that describes the 'should be' process (i.e. a normative process model). Load the file 'normativeModel.pnml' into ProM and apply conformance checking on this process model, and on the full unfiltered original event log.

1. Include a screenshot of the part of the normative process model, with the conformance information projected onto it, that shows where most of the deviations occur.
2. What is the replay fitness (the 'trace fitness' statistic) of the event log on the normative process model?
3. Select the transition 'T06 Determine necessity of stop advice+complete' (on the top left of the model) and discuss its element statistics: how many times is the transition executed correctly and how many times incorrectly?
4. Using the element statistics of transition 'T06 Determine necessity of stop advice+complete', what can you say about the (in)correct execution of this activity?

**Instructions to align the process model with the event log:**

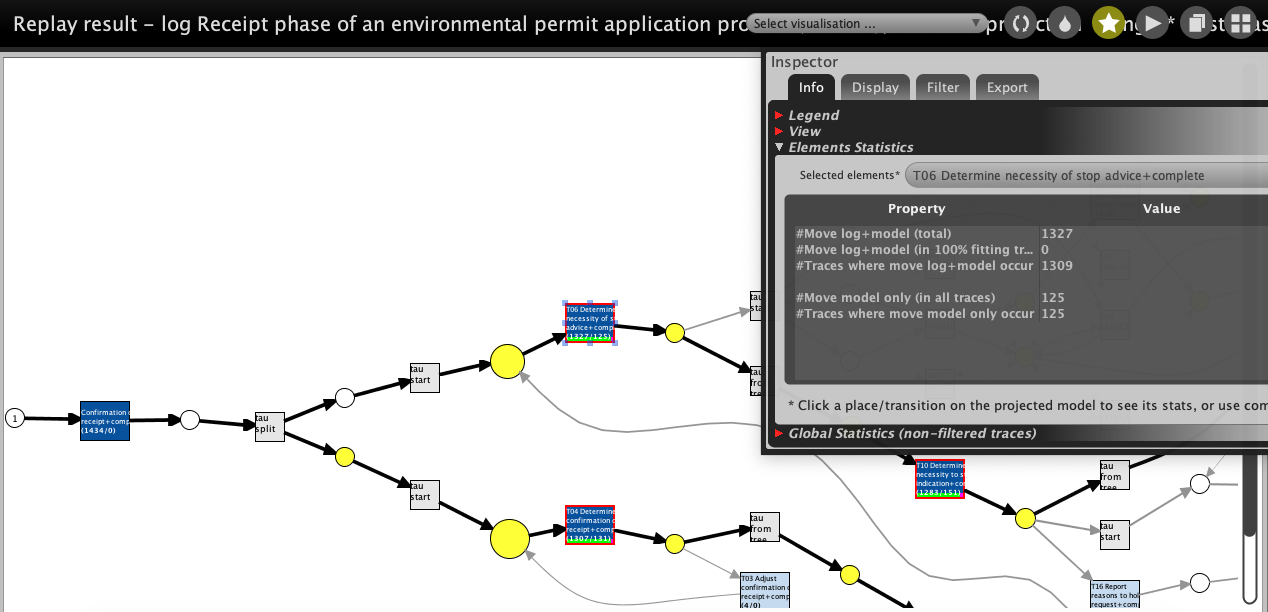
1. Import the normative model using the 'PNML Petri net files' importer.
2. Select the imported normative Petri net and the event log, start the plug-in called ‘Replay a Log on Petri Net for Conformance Analysis’ (not the variant with performance!), and click 'yes' in the 'No Final Marking' pop-up.
3. Select the 'sink' place on the left (note: do not select '0-sink' etc.) and click the button 'Add Place >>' to add the place 'sink' to the candidate final marking list. Now click 'Finish'.
4. Click 'Finish' in the mapping wizard.
5. Click 'No, I've mapped all necessary event classes' to indicate that some events are not present in the normative model.
6. Now click 'Next' and 'Finish'. The normative process model is shown with conformance information projected onto it.

If you followed these instructions exactly you do not need to mention these steps in your answer.

More information regarding this conformance technique is provided in lecture 4.7: 'Aligning observed ****and modeled behavior' (and to a lesser extend in the lectures 4.3 through 4.6).

The replay fitness (the ‘trace fitness’ statistic) of the event log on the normative process model is 0.84254. T10 has the maximum deviations (151). T06 has the minimum (125) amongst the frequent trace variants.

The transition ‘T06 Determine necessity of stop advice+complete’ (on the top left of the model) was tested with 1,434 traces in the event log. Out of those 1,309 (91%) were synchronous moves in both the model & log. Amongst those 1,309 traces, T06 was fired synchronously for 1,327 times (i.e. some traces fired T06 multiple times). For 125 traces, T06 was fired in the model only i.e. T06 was not fired in the event log 125 times when it was supposed to.



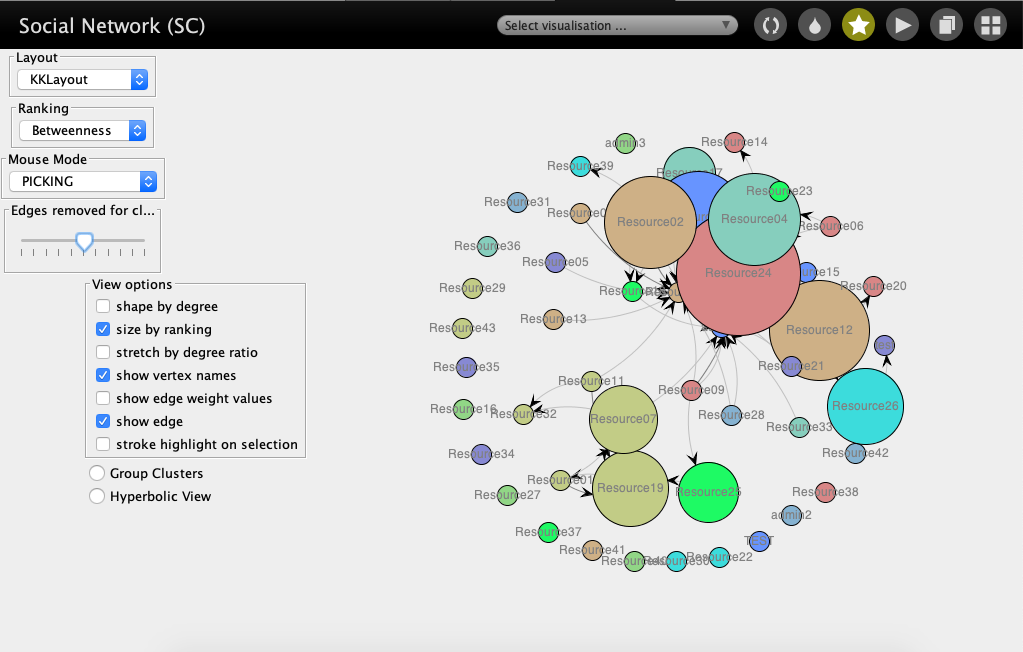
**QUESTION 7**

The final analysis you have to perform on the original event log is a resource analysis, e.g. looking at the user behavior in the event log.

1. Use the plug-in 'Mine for a Subcontracting Social Network'. Note that subcontracting means that if individual *j* frequently executed an activity in-between two activities executed by individual *i*, then individual *i* subcontracted work to individual *j*.

Answer the following question using this view: Can two or more groups of users be distinguished? Explicitly discuss the settings you have used in the resulting visualization.

1. Again use one of the two Dotted Chart plug-ins. For the XDottedChart change the component type to 'org:resource'. If you use the Dotted Chart visualizer change the 'Y Axis Attribute' to 'C: Resource classifier' and the color attribute to 'C: Activity Classifier'. Answer the following two questions using this view:
2. Are all users executing activities from the start of the event log, or are some users joining later?
3. Are users mainly executing particular activities or are most users executing most of the activities?



Settings which I have choosen for resulting visualization

1. Ranking: “Betweenness” it shows ressources acting in-between
2. Edge removed for - to get distingushing colors
3. Size by ranking - to display the in-betweeness by size

We can distinguish 3 types of resource involvement:

1. With big subcontracting involvement, e.g. Resource 24, 02, 12, 08, . . .
2. Several with some subcontracting, e.g. 33, 15, 13, . . . ., admin, . . .
3. Many with no subcontracting at all. 36, 30, 2, . . .



We can conclude:

1. A few resources were active only later, e.g. since Sept. 2011.
2. Some resources are involved only in one or very few different activities.
3. About half of the resources are active most times, the other half only occasionally.
4. Some conducted only one activity at all.