Chapter 15
Cartography and Navigation

Process models can be seen as the "maps" describing the operational processes of organizations. Similarly, information systems can be looked at as "navigation systems" guiding the flow of work in organizations. Unfortunately, many organizations fail in creating and maintaining accurate business process maps. Often process models are outdated and have little to do with reality. Moreover, most information systems fail to provide the functionality offered by today's navigation systems. For instance, workers are not guided by the information system and need to work behind the system's back to get things done. Moreover, useful information such as the "estimated arrival time" of a running case is not provided. Process mining can help to overcome some of these problems.

15.1 Business Process Maps

The first geographical maps date back to the 7th Millennium BC. Since then cartographers have improved their skills and techniques to create maps thereby addressing problems such as clearly representing desired traits, eliminating irrelevant details, reducing complexity, and improving understandability. Today, most geographic maps are digital and of high quality. This has fueled innovative applications of cartography as is illustrated by modern car navigation systems (e.g., TomTom, Garmin, and Navigon), Google Maps, mashups using geo-tagging, etc. There are thousands of mashups using Google Maps, e.g., applications projecting information about traffic conditions, real estate, fastfood restaurants, or movie showtimes onto a selected map. People can seamlessly zoom in and out using such maps and interact with it, e.g., traffic jams are projected onto the map and the user can select a particular problem to see details.

Process models can be seen as the *"business process maps"* describing the operational processes of organizations [138]. Unfortunately, accurate business process maps are typically missing. Process models tend to be outdated and not aligned with reality. Moreover, unlike geographic maps, process models are typically not well understood by end users.

W. van der Aalst, *Process Mining*, DOI 10.1007/978-3-662-49851-4_15

As indicated in Sect. 10.1.1, we suggest *adopting ideas from cartography*. In the remainder of this section, we discuss ways of improving process models inspired by cartographic techniques. Some of these ideas are already supported by existing process mining techniques, others point to further innovations.

15.1.1 Map Quality

Geographical maps are typically of high quality compared to business process maps. For example, the maps used by navigation systems are very accurate, e.g., when driving from Amsterdam to Rome relatively few discrepancies between reality and the map will be encountered.

Process models tend to provide an idealized view on the business process that is modeled. Imagine that road maps would view the real highway system through similar rose-tinted glasses, e.g., showing a road that is not there but that should have been there. This would be unacceptable. However, these are the kind of business process maps used in many organizations. Such a "PowerPoint reality" limits the use and trustworthiness of process models.

In Chap. 8 we showed various conformance checking techniques that can be used as a "reality check" for business process maps. For instance, using replay the fitness of a process model and an event log can be determined. We encountered many real-life processes in which the fitness of the model and the log is less than 0.4. This implies that less than 40% of the behavior seen in reality fits into the model.

Some will argue that road maps are easier to maintain than process models, because a road system evolves at a much slower pace than a typical business process. This is indeed the case. However, this makes it even more important to have accurate up-to-date business process maps!

Besides differences in quality, there are also huge differences in understandability. Most people will intuitively understand geographical maps while having problems understanding process models. The dynamic nature of processes makes things more complicated (cf. workflow patterns [155, 191]). Therefore, the perceived complexity is partly unavoidable. Nevertheless, ideas from cartography can help to improve the understandability of process models.

15.1.2 Aggregation and Abstraction

Figure 15.1 shows a map. The map *abstracts* from less significant roads and cities. Roads that are less important are not shown. A cut-off criterion could be based on the average number of cars using the road per day. Similarly, the number of citizens could be used as a cut-off criterion for cities. For example, in Fig. 15.1 cities of less than 50,000 inhabitants are abstracted from. Maps also *aggregate* local roads and local districts (neighborhoods, suburbs, centers, etc.) into bigger entities.

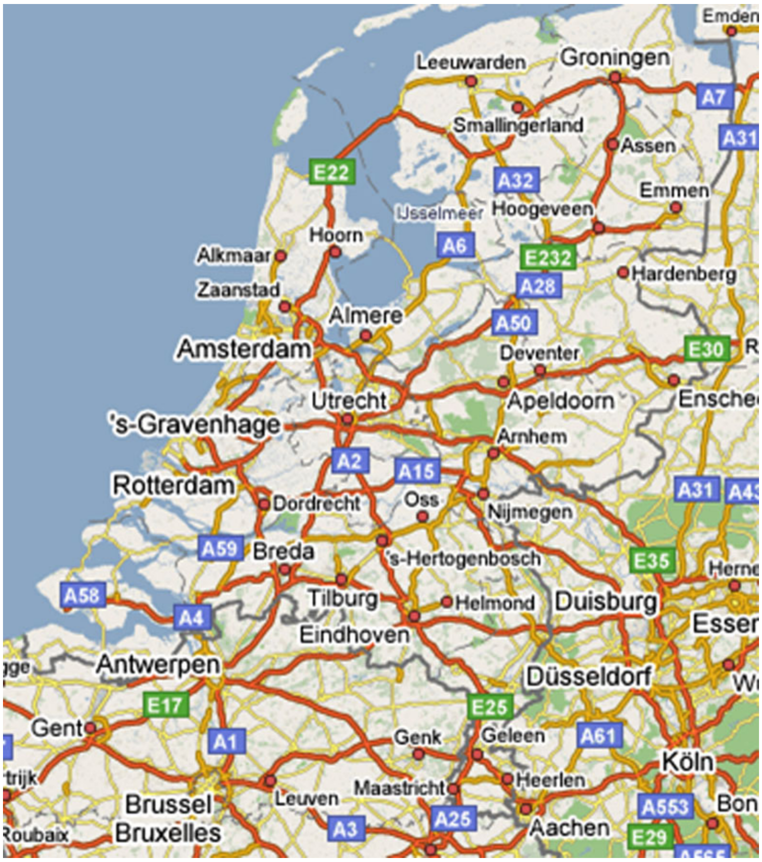


Fig. 15.1 Road map of The Netherlands. The map abstracts from smaller cities and less significant roads; only the bigger cities, highways, and other important roads are shown. Moreover, cities aggregate local roads and local districts

Figure 15.1, for instance, shows Eindhoven as a single dot while it consists of many roads, various districts (Strijp, Gestel, Woensel, Gestel, etc.), and neighboring cities (e.g., Veldhoven). People interested in Eindhoven can look at a city map to see more details.

Process models also need to abstract from less significant things. Activities can be removed if they are less frequent, e.g., activities that occur in less than 20% of completed cases are abstracted from. Also time and costs can be taken into account, e.g., activities that account for less than 8% of the total service time are removed unless the associated costs are more than € 50,000.

Aggregation is important for process mining because many event logs contain low-level events that need to be aggregated into more meaningful activities. In [77] it is shown how frequent low-level patterns can be identified and aggregated. Suppose that $x = \{\langle a, b, c\rangle, \langle a, b, b, c\rangle\}$, $y = \{\langle a, d, e, c\rangle, \langle a, e, d, c\rangle\}$, and $z = \{\langle d, d, d, a\rangle\}$

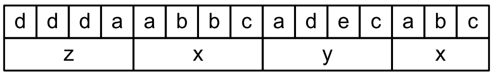


Fig. 15.2 A low-level trace is mapped onto a trace at a higher level of abstraction, e.g., the subsequence $\langle d, d, d, a\rangle$ is mapped onto z

are frequent low-level patterns that represent meaningful activities, e.g., the low-level subsequences a, b, c and a, b, b, c are possible manifestations of activity x. Now consider the low-level trace $\sigma = \langle d, d, d, a, a, b, b, c, a, d, e, c, a, b, c\rangle$. This trace can be rewritten into $\sigma' = \langle z, x, y, x\rangle$ showing the aggregated behavior (see Fig. 15.2). By preprocessing the event log in this way, it is possible to discover a simpler process model. Filtering, as described in Sect. 14.2, can be seen as another form of preprocessing. It is also possible to apply aggregation directly to the graph structure (see fuzzy mining [66] and Sects. 14.2 and 15.1.3).

Aggregation introduces multiple levels. For each aggregate node a kind of "city map" can be constructed showing the detailed low-level behavior. In principle there can be any number of levels, cf. country maps, state maps, city maps, district maps, etc.

15.1.3 Seamless Zoom

There may be different geographic maps of the same area using different scales. Moreover, using electronic maps it is possible to seamlessly zoom in and out. Note that, while zooming out, insignificant things are either left out or dynamically clustered into aggregate shapes (e.g., streets and suburbs amalgamate into cities). Navigation systems and applications such as Google Maps provide such a seamless zoom. Traditionally, process models are static, e.g., it is impossible to seamlessly zoom in to see part of the process in more detail. To deal with larger processes, typically a *static hierarchical decomposition* is used. In such a hierarchy, a process is composed of subprocesses, and in turn these subprocesses may be composed of smaller subprocesses.

Consider, for example, the WF-net shown in Fig. 15.3. The WF-net consists of atomic activities $(a, b, \ldots, l)$ partitioned over three subprocesses x, y, and z. The overall process is composed of these three subprocesses. Figure 15.4 shows the top-level view of this composition. The semantics of such a hierarchical decomposition is the "flattened" model, i.e., subprocesses at the higher level are recursively replaced by their inside structure until one large flat process model remains (in our example there are only two levels).

Figures 15.3 and 15.4 show the limitations of hierarchical decomposition. At the highest level one needs to be aware of all interactions at the lower levels. The reason is that higher levels in the decomposition need to be consistent with the lower levels, e.g., because there is a connection between activity l and activity b at the lower level, there also needs to be a connection between z and x at the higher level.

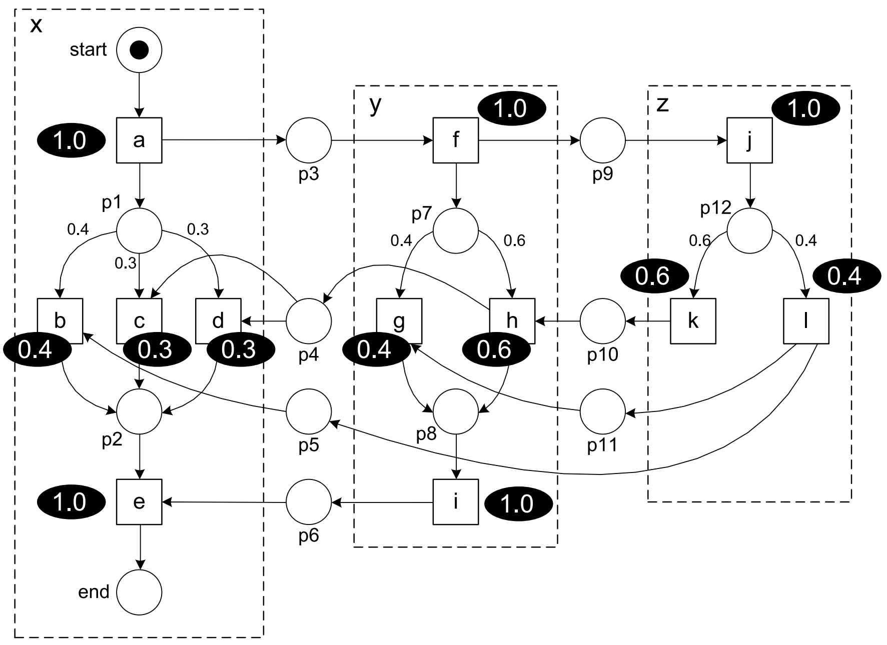
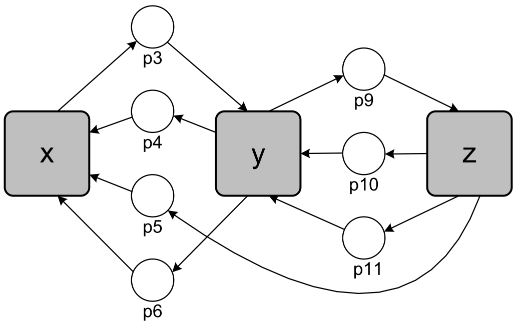


Fig. 15.3 A WF-net consisting of 12 atomic activities partitioned over three subprocesses x, y, and z. The average frequency of each activity is shown. For instance, activity h is executed for 60% of the cases

Fig. 15.4 Top-level view on the hierarchical WF-net shown in Fig. 15.3



This is not only the case for WF-nets, but holds for the hierarchy constructs in other languages such as BPMN, YAWL and EPCs. From a design point of view, hierarchical decomposition makes perfect sense. When designing a system it is important to ensure consistency between different levels and the possibility to "flatten" models provides clear execution semantics.

However, when *viewing* a process model it is important to be able to zoom out to see fewer details and zoom in to see more details. This implies that the view is not static, i.e., activities should not be statically bound to a particular level chosen at design time. Moreover, when abstracting from infrequent low-level behavior the

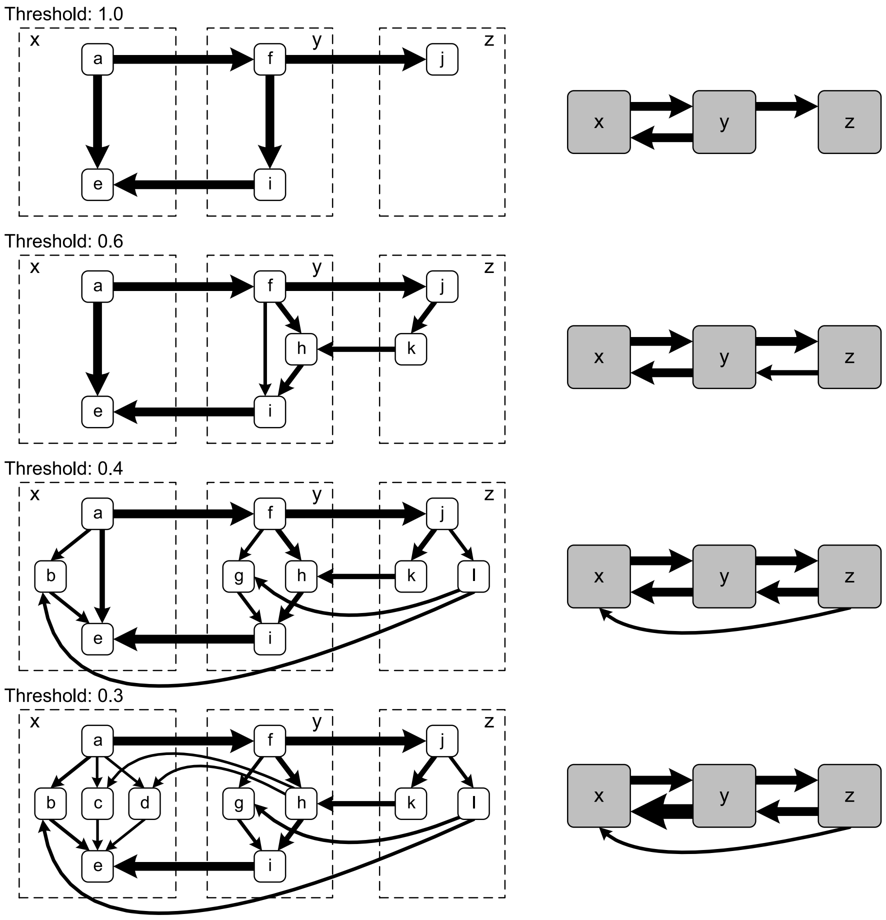


Fig. 15.5 The process specified by the WF-net of Fig. 15.3 viewed at different levels of abstraction. An activity and its corresponding connections are removed if the transition is less frequent than the threshold. Both the atomic view (*left*) and the aggregate view (*right*) are shown for four threshold values (0.3, 0.4, 0.6, and 1.0)

corresponding connections at higher levels should also be removed. For instance, if activity l is very infrequent, it is not sufficient to hide it at a lower level: the connection between z and x (i.e., place $p5$) should also be removed.

Figure 15.5 illustrates how processes can be viewed while taking into account the frequencies of activities. As shown in Fig. 15.3, activities a, e, f, i, and j have a frequency of 1, i.e., they are executed once for each case. Activities h and k are executed for 60% of the cases, and activities b, g, and l are executed for 40% of the cases. Activities c and d are least frequent and are executed for only 30% of the cases. Assume that we would like to seamlessly simplify the model by progressively leaving out more activities based on their frequencies. Figure 15.5 shows four different levels. Here, we abstract from the detailed process logic and only show activities and their connections. Moreover, we show the intensity of connections by proportionally varying the width of the arcs. If the threshold is set to 0.3, then all activities are included. When the threshold is increased to 0.4, then activities c and d and their connections disappear. When the threshold is increased to 0.6, also activities b, g, and l and their connections disappear. If the threshold is set to 1, then only the most frequent activities are included. The left-hand side of Fig. 15.5 shows atomic activities and their relations. The right-hand side of the figure shows the connections if we assume that the activities are aggregated as shown in the original WF-net (cf. Fig. 15.3). It is important to note that the connection between z and x disappears when the threshold is higher than 0.4. If we abstract from the infrequent activities b and l, then we should also remove this connection. For the same reason the connection between z and y is not shown when the threshold is set to 1.

Figure 15.5 shows how one can seamlessly zoom in and zoom out to show more or less detail. This is very different from providing a static hierarchical decomposition and showing a particular level in the hierarchy as is done by the graphical editors of BPM systems, WFM systems, simulation tools, business process modeling tools, etc.

Thus far we assumed a static partitioning of atomic activities over three subprocesses. Depending on the desired view this partitioning may change. To illustrate this, we use an example event log consisting of 100 cases and 3730 events. This event log contains events related to the reviewing process of journal papers. Each paper is sent to three different reviewers. The reviewers are invited to write a report. However, reviewers often do not respond. As a result it is not always possible to make a decision after a first round of reviewing. If there are not enough reports, then additional reviewers are invited. This process is repeated until a final decision can be made (accept or reject). Figure 15.6 shows the process model discovered by the α-algorithm.

The α-algorithm does not allow for seamlessly zooming in and out. One would need to filter out infrequent activities from the log and subsequently apply the α-algorithm to different event logs. The *Fuzzy Miner* of ProM allows for seamlessly zooming in and out as is shown in Fig. 15.7 [65, 66]. The three fuzzy models shown in Fig. 15.7 are all based on the event log also used by the α-algorithm. Figure 15.7(a) shows the most detailed view. All activities are included. The color and width of the connections indicate their significance (like in Fig. 15.5). Figure 15.7(b) shows the most abstract view. The decision activity is typically executed multiple times per paper. Therefore, it is most frequent. The other 18 activities are partitioned over 4 so-called cluster nodes. Each cluster node aggregates multiple atomic activities. Using a threshold similar to the one used in Fig. 15.5, the Fuzzy Miner can seamlessly show more or less details. Figure 15.7(c) shows a model obtained using an intermediate threshold value. The top-level model shows the six most frequent activities. The other activities can be found in the three cluster nodes. Figure 15.7(d) shows the inner structure of an aggregate node consisting of 10 atomic activities. Note that the inner structure of an aggregate node shows the connections to nodes at the higher level.

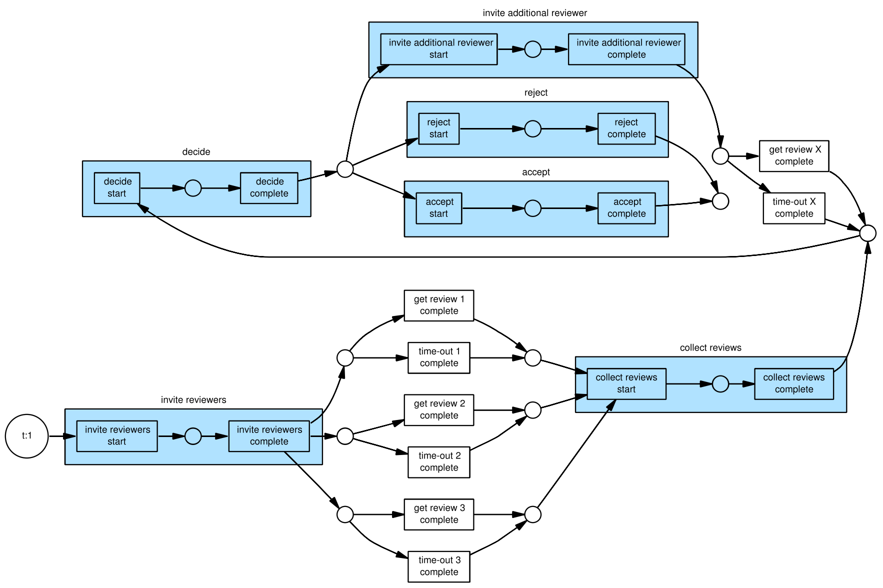


Fig. 15.6 WF-net discovered using the α-algorithm plug-in of ProM. An event log consisting of 100 cases and 3730 events was used to create the model. All activities are shown in the discovered model

When zooming out using Google maps, less significant elements are either left out or dynamically clustered into aggregate shapes. For example, streets and suburbs amalgamate into cities. This is similar to the zoom functionality provided by ProM's Fuzzy Miner as was demonstrated using Fig. 15.7. Note that in this particular example activities are aggregated and not removed. The Fuzzy Miner has many parameters that allow the user to influence the resulting model. Using different settings of these parameters it is also possible to abstract from activities (i.e., remove them) rather than aggregating them. Activities can also be removed by filtering the event log before applying a discovery algorithm (see Sect. 14.2).

15.1.4 Size, Color, and Layout

Cartographers not only eliminate irrelevant details, but also use colors to highlight important features. For instance, the map shown in Fig. 15.1 emphasizes the importance of highways using the color red. Moreover, graphical elements have a particular size to indicate their significance, e.g., the sizes of lines and dots may vary. For instance, in Fig. 15.1 the size of a city name is proportional to the number of citizens, e.g., Zaanstad is clearly smaller than Amsterdam. Geographical maps also have a clear interpretation of the x-axis and y-axis, i.e., the layout of a map is not arbitrary as the coordinates of elements have a meaning.

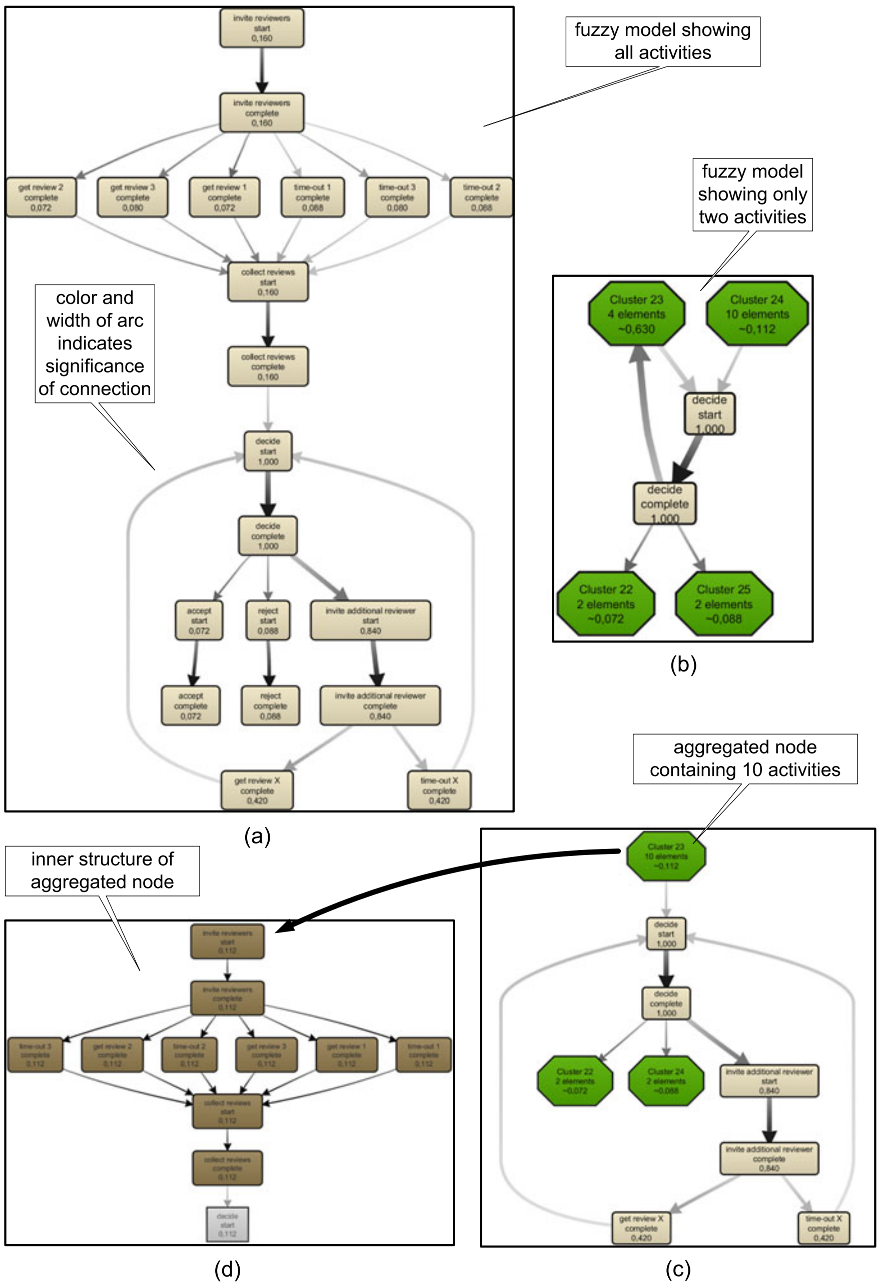

fuzzy model showing all activities
fuzzy model showing only two activities
color and width of arc indicates significance of connection
aggregated node containing 10 activities
inner structure of aggregated node
invite reviewers start 0,160
invite reviewers complete 0,160
get review 2 complete 0,072
get review 3 complete 0,080
get review 1 complete 0,072
time-out 1 complete 0,088
time-out 3 complete 0,080
time-out 2 complete 0,088
collect reviews start 0,160
collect reviews complete 0,160
decide start 1,000
decide complete 1,000
accept start 0,072
reject start 0,088
invite additional reviewer start 0,840
accept complete 0,072
reject complete 0,088
invite additional reviewer complete 0,840
get review X complete 0,420
time-out X complete 0,420
Cluster 23 4 elements ~0,630
Cluster 24 10 elements ~0,112
Cluster 22 2 elements ~0,072
Cluster 25 2 elements ~0,088
Cluster 23 10 elements ~0,112
Cluster 24 2 elements ~0,088
(a)
(b)
(c)
(d)

Fig. 15.7 Three business process maps obtained using ProM's Fuzzy Miner. The most detailed fuzzy model (**a**) shows all activities. The least detailed fuzzy model (**b**) shows only two activities; all other activities are aggregated into so-called "cluster nodes". The third fuzzy model (**c**) shows six activities. For one of the aggregate nodes, the inner structure is shown (**d**)

All of this is in stark contrast with mainstream process models. The x-axis and y-axis of a process model have no meaning, e.g., the layout of the WF-net shown in Fig. 15.6 was generated automatically without assigning any semantics to the positions of activities. Although modeling tools allow for using colors, the color typically has no semantics. The different types of model elements (e.g., activities, gateways, events, connectors, and places) typically have a default color. Moreover, the size of a model element also has no semantics. Typically all elements of a particular type have the same size.

Because size, color, and layout are not employed when creating business process maps, the result is less intuitive and less informative. However, ideas from cartography can easily be incorporated in the construction of business process maps. Some examples:

- The *size of an activity* can reflect its frequency or some other property indicating its significance (e.g., costs or resource use).
- The *color of an activity* can reflect the mean service time of the activity. For example, activities that take longer than average are colored red whereas short running activities are colored green.
- The *width of an arc* can reflect the importance of the corresponding causal dependency.
- The *coloring of arcs* can be used to highlight bottlenecks.
- The *positioning of activities* can have a well-defined meaning. Similar to swimlanes the y-axis could reflect the role associated to an activity. Similar to a Gantt chart, the x-axis could reflect some temporal aspect.

It is important to use these conventions in a consistent manner across different maps.

15.1.5 Customization

The same geographic area is typically covered by many different maps. There are different maps depending on the type of activity they intend to support, e.g., bicycle maps, hiking maps, and road maps. Obviously, these maps use different scales. However, there are more differences. For instance, a bicycle map shows bicycle paths that are not shown on motorists' map.

Figure 15.7 illustrates that multiple views can be created for the same reality captured in an event log. In earlier chapters we already showed that there is no such thing as *the* process model describing a process. Depending on the questions one seeks to answer, a customized process model needs to be created. In Sect. 6.4.4, we referred to this as taking a "2-D slice of a 3-D reality". The same process can be viewed from different angles and at different levels of granularity. For noisy event logs one may prefer to focus on just the main behavior or also include less frequent behavior. For example, from an auditing point of view the low frequent behavior may be most interesting.

15.2 Process Mining: TomTom for Business Processes?

After comparing geographic maps with business process maps, we now explore the analogy between navigation systems and information systems. Section 10.1.3 already mentioned navigation activities in the context of the refined process mining framework (cf. Fig. 10.1) By establishing a close connection between business process maps and the actual behavior recorded in event logs, it is possible to realize TomTom-like functionality. Analogous to TomTom's navigation devices, process mining tools can help end users (a) by navigating through processes, (b) by projecting dynamic information on process maps (e.g., showing "traffic jams" in business processes), and (c) by providing predictions regarding running cases (e.g., estimating the "arrival time" of a case that is delayed) [138].

15.2.1 Projecting Dynamic Information on Business Process Maps

The navigation systems of TomTom can be equipped with so-called "LIVE services" (cf. www.tomtom.com) showing traffic jams, mobile speed cameras, weather conditions, etc. This information is projected onto the map using current data.

In Chap. 9, we showed that a tight coupling between an event log and a process model can be used to extend process models with additional perspectives, e.g., highlighting bottlenecks, showing decision rules, and relating the process model to organizational entities. The same coupling can also be used to visualize "pre mortem" event data. Information about the current state of running cases can be projected onto the process model.

The idea is analogous to mashups using geo-tagging (e.g., Panoramio, HousingMaps, Funda, and Flickr). Many of these mashups use Google Maps. Consider, for example, the map shown in Fig. 15.8. Prospective customers can visit the site of Funda to look for a house that meets particular criteria. Information about houses that are for sale are projected onto a map. Figure 15.8 shows the houses that are for sale in Hapert. Figure 15.9 shows another example. Now the map shows traffic jams. Both maps are dynamic, i.e., the information projected onto these maps changes continuously.

Both historic and current event data can be used to "breathe life" into otherwise static business process maps. Similar to the visualization of traffic jams in Fig. 15.9, "traffic" in business processes can be visualized. Besides process maps, one can also think of other maps to project information on. Consider, for example, the social networks shown in Figs. 9.6 and 9.7. Work items waiting to be handled can be projected onto these models, e.g., cases that are waiting for a decision by a manager are projected onto the manager role. Some work items also have a geographic component, e.g., a field service engineer could be provided with a map like the one in Fig. 15.8 showing the devices that need maintenance. It is also possible to project work items onto maps with a temporal dimension (Gantt charts, agendas, etc.). For instance, a surgeon could view scheduled operations in his agenda. Hence, a variety of maps covering different perspectives can be used to visualize event related data.

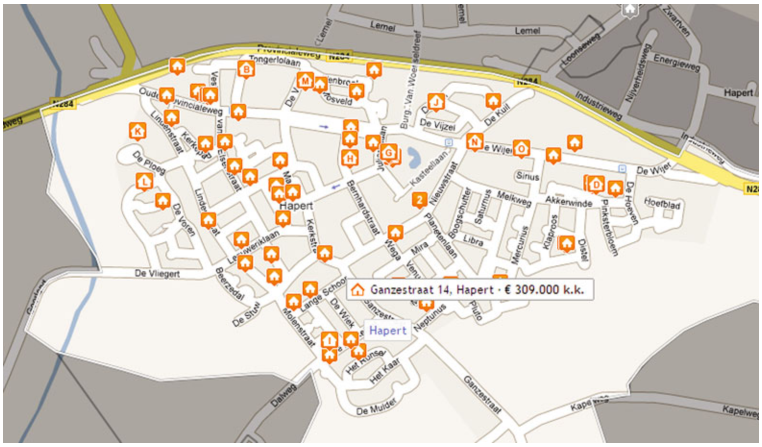


Fig. 15.8 Funda allows users to view maps with houses for sale that meet particular criteria, e.g., constraints related to size, volume, and pricing. The map shows the 53 houses for sale in the Dutch town Hapert

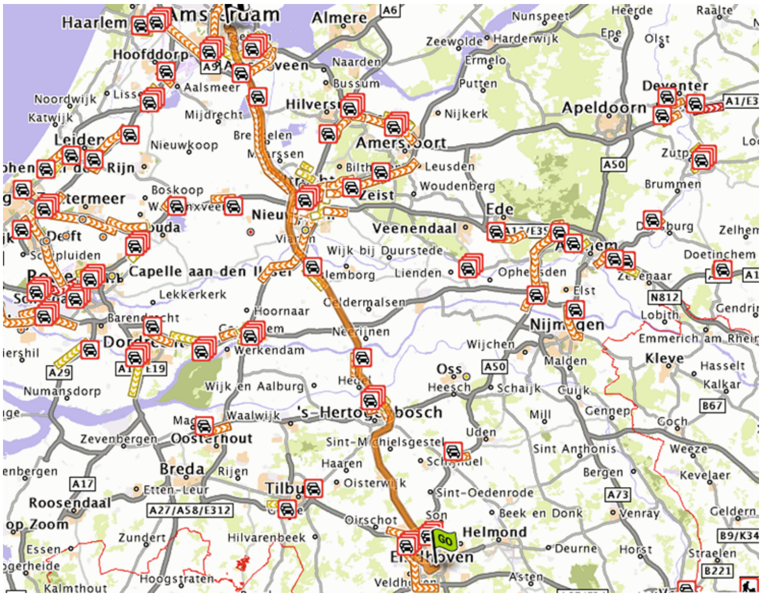


Fig. 15.9 Road map showing traffic jams: the car icons indicate problem spots and congested roads are highlighted. Modern navigation systems show such maps and, based on real-time traffic information, alternative routes are suggested to avoid bottlenecks

The YAWL system [41, 150] provides a visualization framework able to map pending work items and resources onto various maps, e.g., geographic maps, process maps, and organizational maps. YAWL also defines various distance notions

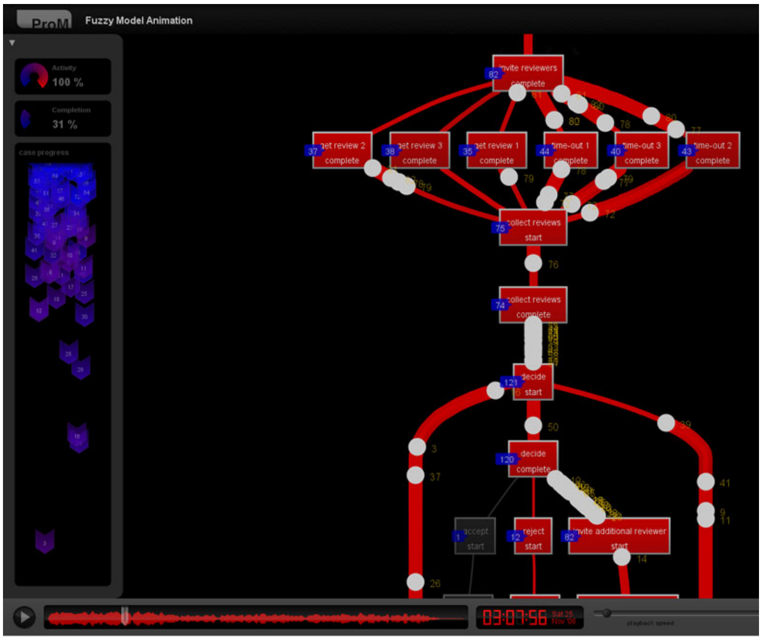


Fig. 15.10 The fuzzy model discovered earlier (cf. Fig. 15.7(a)) is used to replay the event log. The animation reveals the problem that many reviewers do not provide a report in time. As a result the editor of the journal cannot make a final decision and needs to invite additional reviewers. There is long queue of work items waiting for a decision and many pending invitations

based on these maps, for instance, a field service engineer can see the work items closest or most urgent.

From business process maps to business process movies
Once events in the log can be related to activities in the process model, it is possible to replay history on a case-by-case basis. This was used for conformance checking and model extension. Now we go one step further; we do not consider an individual case but all relevant cases at the same time. Assuming that events have a timestamp, all events in the log can be globally ordered, i.e., also events belonging to different cases can be sorted. After each event, the process is in a particular global state. One can think of this state as a *photograph* of the process. The state can be projected onto a business process map, a geographic map, or an organizational map. Since such a photograph is available after each event, it is also possible to create a *movie* by simply showing one photograph after another. Hence, it is possible to use event logs to create a "business process movie". Fig. 15.10 shows an example using ProM's Fuzzy Miner [65, 66]. The event log and the fuzzy model are converted into

an animation. The dots visible in Fig. 15.10 are moving along the arcs and refer to real cases. Such a business process movie provides a very convincing means to show problems in the as-is process. Unlike simulation, the animation shows reality and people cannot dismiss the outcomes by questioning the model. Therefore, business process movies help to expose the real problems in an organization.

15.2.2 Arrival Time Prediction

Whereas a TomTom device is continuously showing the *expected arrival time*, users of today's information systems are often left clueless about likely outcomes of the cases they are working on. This is surprising as many information systems gather a lot of historic information, thus providing an excellent basis for all kinds of predictions (expected completion time, likelihood of some undesirable outcome, estimated costs, etc.). Fortunately, as shown in Sect. 10.4, event logs can be used build predictive models.

The annotated transition system [164, 167] described in Sect. 10.4 can be used to predict the remaining flow time of a running case. The transition system is constructed using an event log L and a state representation function $l^{state}()$ or obtained by computing the state-space of a (discovered) process model. By systematically replaying the event log, the states are annotated with historic measurements. The mean or median of these historic measurements can be used to make predictions for running cases in a particular state. Each time the state of a case changes, a new prediction is made for the remaining flow time. Clearly, this functionality is similar to the prediction capabilities of a navigation device. Moreover, using different annotations, other kinds of predictions can be made. For instance, the transition system can be annotated with cost information to predict the total or remaining costs. Similarly, the outcome of a process or occurrence of an activity can be predicted.

Alternative approaches based on regression analysis, short-term simulation, or decision tree learning can be used to predict properties such as the remaining flow time of a running case. This illustrates that process mining can be used to extend information systems with predictive analytics.

15.2.3 Guidance Rather than Control

Car navigation systems provide *directions* and *guidance without controlling* the driver. The driver is still in control, but, given a goal (e.g. to get from A to B as fast as possible), the navigation system recommends the next action to be taken. In Sect. 10.5 we showed that predictions can be turned into recommendations. Recommendations are given with respect to a goal, e.g., to minimize costs, to minimize the remaining flow time, or to maximize the likelihood of success. Such a goal is operationalized by defining a performance indicator that needs to be minimized or maximized. For every possible next action, the value of the performance indicator is predicted. This information is used to rank the possible actions and thus recommend the next step to be taken (cf. Fig. 10.12).

Recommendations based on process mining allow for systems that are flexible but also supporting operational decision making. Today's information systems typically do not provide a good balance between flexibility and support. The system is either restricting people in their actions or not providing any guidance. BPM systems offering more flexibility (e.g., case handling systems like BPM|one or declarative workflow systems like Declare), can be extended with a recommendation service based on process mining techniques [164].

The TomTom metaphor illustrates that many information systems lack functionality present in today's navigation devices [138]. However, high-quality process models tightly coupled to event logs enable TomTom-like functionalities such as predicting the "arrival time" of a process instance, recommending the next activity to be executed, and visualizing "traffic jams" in business processes.