Automata and Petri Net Models for Visualizing and Analyzing Complex Questionnaires – A Case Study –

Heiko Rölke

Deutsches Institut für Internationale Pädagogische Forschung
(German Institute for International Educational Research)
Solmsstraße 75, 60486 Frankfurt, Germany
roelke@dipf.de

Abstract. Questionnaires for complex studies can grow to considerable sizes. Several hundred questions are not uncommon. In addition, routings are used to distinguish between question paths for different respondents. This leads to the question of how to ensure validity and other important properties.
We examine this question for a case with even more demanding side conditions: An important part of the OECD study "PIAAC" (Programme for the International Assessment of Adult Competencies) is a background questionnaire (BQ) containing more than 400 questions. This BQ has to be adapted by all participating countries. Nevertheless, integrity of the overall system has to be ensured.

Keywords: automata, Petri nets, questionnaires, analysis

1 Motivation and Overview

Large scale studies in psychology, sociology, and for many other purposes try to find out characteristics of complete populations or at least of big parts of a population. International studies often aim at comparing the complete population of one country to that of another country. The well-known PISA study of the OECD, as an example, aims at comparing *all* students of age 15 worldwide. This is done by examining representative samples in each country that participates in PISA.

To be able to compare one first has to find out some background information about the people that are compared. This is mainly done by asking those people questions. Larger chunks of questions grouped together in order to find about the *background* of the surveyed people are called *background questionnaires*, or BQ in short.

Recent Advances in Petri Nets and Concurrency, S. Donatelli, J. Kleijn, R.J. Machado, J.M. Fernandes (eds.), CEUR Workshop Proceedings, ISSN 1613-0073, Jan/2012, pp. 317–329.

The author of this paper was involved in the definition, implementation, national adaptation, and deployment of the BQ for the OECD PIAAC study.[1] The OECD is the "Organization for Economic Co-Operation and Development", see [5] for details. Among many other activities, the OECD is well-known for organizing world-wide comparability studies, like the PISA study. PISA [6] is the abbreviation of "Programme for International Student Assessment". The PIAAC study, "Programme for the International Assessment of Adult Competencies" can be seen as an extension of the PISA study for adults. It aims at finding out about skills needed by adults in order to be successful in everyday work life. See [7] for details about the PIAAC study. PIAAC is carried out by 24 countries all over the world (participating countries are located in North and South America, Europe, Asia and Oceania).

1.1 Background Questionnaire Properties

There is no exact definition of what a BQ is. It is therefore not possible to exactly determine properties that have to be valid for each and every BQ. Naively, it is just a bunch of questions that an interviewer has to present to an interviewee. In practice, in discussions with psychologists, sociologists, or other questionnaire practitioners, certain universally agreed principles and best practices become clear. From this starting point, desirable properties can be derived. Nevertheless, it is not possible in the moment to definitely define and answer all related questions. We strive for more general validity, though.

A BQ usually has one single entry or starting point, the first *item* or *question*. In practice, this is often a hidden item, where predefined data is imported. An example: One often knows the name of the interviewee in advance. Within the BQ, there may be many different paths through the question pool, often depending on previously entered data or chosen randomly. An item that is intended to be the last question of a BQ is called an *end item*. Again, this may be a visible item (a question) or a hidden item not visible to the interviewer. While there often only is one end item, for example thanking the interviewee for time and patience or, more technically, exporting the assembled data, this is not a standard requirement of a BQ.

Due to practical considerations, there often is the possibility to pause an interview or to break it off. While pausing has no implications for the structure, a break-off means that any item can be an end item or has a connection to an end item.

The normal flow through a BQ should not result in a dead end. A dead end is an item that was not considered to be an end item. Other requirements are more on the semantic side. Each possible question sequence has to make sense semantically. On the other hand, each desired or planned sequence has to be possible, e.g. by entering appropriate answers.

[1] The work was done in the international consortium responsible for implementing, deploying, and analyzing the study, led by ETS [3] in Princeton, USA. Most of the implementation work on the BQ was done by the CRP Henry Tudor [2] in Luxembourg.

1.2 Overview

The rest of the paper is structured as follows: In Section 2 we give an overview on the BQ of the PIAAC study. We also give examples of the format in that the BQ is defined. Following up on that we develop first simple models for the PIAAC BQ in Section 3. The models are put into practice in Section 4. They are used to gain quite some insight and find errors, but are not sufficiently powerful to represent all important aspects of our application. So we carry on in Section 5 with more powerful Petri net models that allow for more sophisticated analysis. We conclude in Section 6 with an outlook on further work and possible generalizations.

2 The PIAAC Background Questionnaire

The PIAAC study mainly consists of two important parts: a background questionnaire (BQ) and cognitive tests (cognitive items, CI). Both parts are embedded into an overall workflow that controls all parts of the survey. This workflow is implemented in the same way as the BQ.

The PIAAC BQ starts with general questions about the interviewee to find out whether he is suited to take part in the survey or not. Afterwards questions in different categories are asked, grouped together in blocks. Examples for such blocks are questions about the educational background, skills needed in everyday work, and questions about private life related to work skills. In order to shorten the overall interview time, parts of the blocks are arranged in a rotated design so that not all interviewees are asked the same questions. Another example of inter-block routing is that certain blocks are not administered if the requirements for asking these questions are not fulfilled, e.g. questions about current work in case of an unemployed interviewee.

In addition to the inter-block routing, complex routing is used within blocks to administer the right questions. A good example for such a routing are questions about the education of an interviewee: If an interviewee has never been to an university it is useless to ask questions about academic degrees. Respective questions should be skipped. Another typical situation includes loops: One might be interested in the degree of skills related to foreign languages. To accommodate speakers fluent in multiple languages, some kind of cycle or loop is needed.

The code example in Figure 1 shows an example of a questionnaire item with a free text entry. The XML syntax is not important here.[2] The item group that is defined in the code snipped defines a single item, i.e. one question is administered. The item has a unique identifier (ID), instruction text and answering possibilities. In this case a free text entry of length 12.

The second code example in Figure 2 shows a routing with two possible targets. This is a hidden item, i.e. an item that is not displayed but used internally.

[2] The XML syntax of the PIAAC BQ has been specially designed for this purpose. At the time of writing of this paper only limited support like editors or visualizers is available.

```
<itemGroup id="CI_PERSID" responseCondition="ALL" layout="list">
  <item id="CI_PERSID">
    <instruction>Please enter the sampled person ID</instruction>
    <responses layout="radioButton">
      <response code="00" freeTextEntry="true"
         freeTextEntrySize="12" > Sampled Person ID:[FTE]</response>
    </responses>
  </item>
</itemGroup>
```

Fig. 1. BQ code example: free text entry

```
<itemGroup id="CI_skip-C-200Rule" layout="list"
   responseCondition="ALL" hidden="true">
<item id="CI_skip-C-200Rule"/>
<routing>
  <condition>
    <operator type="equal">
      <variable name="CI200Rule"/>
      <constant>NI</constant>
    </operator>
  </condition>
  <then>
    <goto itemGroup="CI200Rule"/>
  </then>
  <else>
    <goto itemGroup="CI_start"/>
  </else>
</routing>
</itemGroup>
```

Fig. 2. BQ code example - conditional routing

The routing is conditional. It is based on the value of the variable `CI200Rule`. Based on this variable, the BQ jumps to item `CI200Rule` or `CI_start`.

Each variable can only be written once. There is a one-to-one relationship between an item and a variable. The variable has the same name as the item where it is initialized and written. Afterwards the variable can only be read, not changed or deleted. There is a notable exception to this rule: It is possible to go back in the questionnaire, for example in case of an error noticed later on. If this is done, the variables connected to the items eventually asked again can also be written again.

The PIAAC BQ together with the overall survey workflow contains more than 600 items. It has one single start item and one single end item. It is possible to break-off the interview at many items, but not all. Break-off leads to a special item that asks for the reason for the break-off.

3 BQ Modeling

A basic modeling strategy for background questionnaires is relatively straightforward. Items (and/or item groups) can be modeled for example as states of a finite automata. Going from one question to the other is a matter of transition from one state the the next. Routing can be modeled as conflicting state transitions. We will now have a closer look at this idea and discuss whether it is sufficient below.

3.1 Automata models

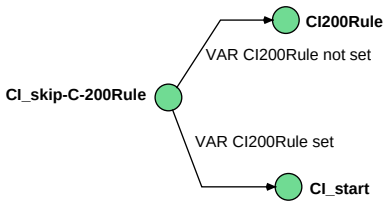

Fig. 3. Finite automata for code in Fig. 2

Figure 3 illustrates the idea of an automata model for the BQ. The automata implements the BQ code example of Figure 2. State `CI_skip-C-200Rule` is connected to the states `CI200Rule` and `CI_start`. The actual transition depends on the variable `CI200Rule` as described above. This data dependency causes problems with this basic model. We will come back to this problem later on.

The benefit even of such a simple formal model is twofold: Once a questionnaire is transformed to a finite automaton, automatic as well as manual

inspection is possible. Automatic inspection can check important properties like connectedness and reachability of the final state(s). Manual inspection is enabled by using a graphical tool that displays the complete BQ. This allows for a much more convenient way of getting an overview of the BQ. It is nearly impossible to follow all routings in the sequential XML format, even if this is supported by an appropriate style sheet (XSLT, see [19]) using links and an overview frame. See Figure 8 to get an idea of the complexity of the BQ. Note that this figure only shows a small part of the overall questionnaire. The model shown in the figure is implemented as a Petri net, not an automaton.

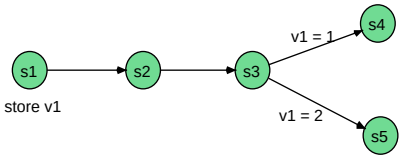

Fig. 4. Data dependent routing

Figure 4 illustrates a general problem with the simple modeling approach. In state `s1` the variable `v1` is written. Afterwards another state (`s2`) is reached and then `s3`. Only in this state the value of `v1` is read again to determine which state (`s4` or `s5`) should be reached next. This means that the variable is used non-locally. While such a situation is common in questionnaires, it is not possible to model a non-local usage of a variable in an ordinary finite automata.[3]

3.2 Petri net models

To overcome the problem of non-local variable usage illustrated above, we remodel the very same BQ part as a Petri net.[4] This can be seen in Figure 5.

As we can see in the figure, the problem can easily be overcome. Items, previously modeled as states in the automaton, are now modeled as places of the Petri net. Transitions have been introduced between the states/places. The places can be seen as the static part, e.g. question or instruction. The transitions are the dynamic part, e.g. the answer given to the respective question and/or the stored variable. Depending on the values stored and retrieved in the variables, the resulting net can be a Place-/Transition net or a colored Petri net.

Place-/Transition nets are possible for variables with restricted (=finite) domains. Luckily, this type is most commonly used in BQs. The vast majority of

[3] This is not completely true, because for variables with finite domains it would be possible to enumerate all reachable states for all values of all variables. Nevertheless, such a model would be hard to read and not very useful.

[4] Petri nets are not an arbitrary choice. They offer various advantages: graphical representation, formal analysis, tool support.

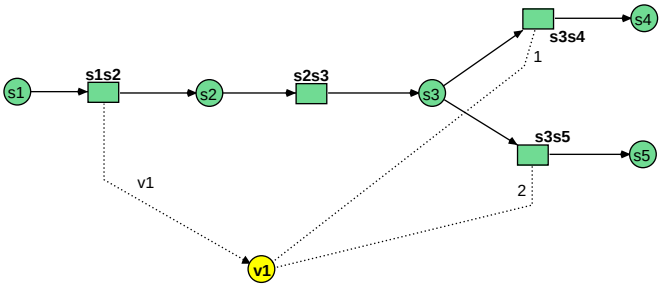

Fig. 5. Data dependent routing of Fig. 4 as PN model

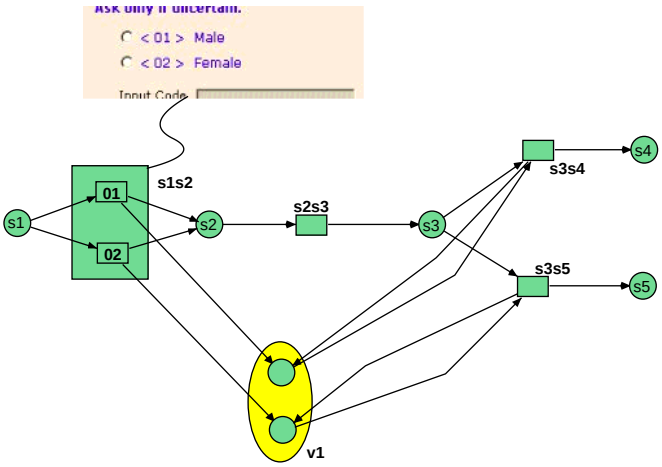

Fig. 6. Refinement of Fig. 4 as PT net

questions is of a single or multiple-choice type. Such a question is illustrated in Figure 6. The first question in this example is about the gender of the interviewee. Only two answers are possible. This can be modeled by a refinement of the net of Figure 5, as shown above. The resulting net is a P/T-net and can be analyzed using the respective tools.

For free text variables or numbers such a modeling would not be possible. Instead, we can use colored nets that support high-level data structures for places and variables directly. While such models are more difficult to analyze they offer other advantages. We will stick to P/T-nets for the moment and come back to the advanced net models later on in Section 5.

4 Modeling and Analysis for PIAAC

The definition, implementation, national adaptation, and deployment of the PIAAC BQ was driven by a high time pressure. Pre-existing questionnaire parts had to be combined and extended. A compromise had to be found that was (a) not too long, (b) implementable world-wide - both a cultural and a political challenge, and (c) able to gather enough data to give answers to the grounding questions of PIAAC. Therefore the work on the BQ started using a semi-structured approach (printable and human-readable Excel sheets), to be able to quickly disseminate all intermediate versions and get feedback. Only late in the process, this was transformed to a well-defined XML format. Therefore also the work on the formal analysis of the BQ started late and is not completely done yet.

The first attempt to get some insight into the BQ structure handled the BQ as a graph. Only an internal model was built, without any graphical representation. Variables were neglected, only the control flow was mapped. From this simple model some important insights were possible: We found dangling routings (jumps to undefined items, e.g. due to spelling mistakes), duplicate item names and isolated nodes - items that could never be reached. On the other hand, it turned out to be quite tedious to verify and analyze the error reports of the first analyzer because of the lack of a graphical representation.

Therefore we implemented another approach targeting on Petri nets. This formalism was chosen to be able to benefit from the advanced set of tools available, allowing to visually inspect a net and formally analyse it at the same time. Our work greatly benefited from the existence and widespread support of the PNML standard - see [8] for an overview or the web site [16] for more information. The usage of PNML allowed to implement the modeling process – modeling a Petri net that represents a specific BQ – as an XSLT transformation.

The first attempt to do so replicated the graph analyzer mentioned above. Variables were neglected, all routing possibilities were handled equally without interpreting the routing conditions. This resulted in a PNML net definition file only containing places, transitions, and arcs. An example can be found in Figure 7. Note that `[...]` means the omitting of plenty of PNML code.

```
<?xml version="1.0" encoding="UTF-8"?>
<pnml>
  <net id="piaac-BQ-DE-001" type="piaac-analyse">
[...]
   <place id="B_C02b1DE2b">
      <name>
        <text>B_C02b1DE2b</text>
      </name>
    </place>
[...]
    <transition id="t_B_C02b1DE2b_B_Q02b2DE2_32">
      <name>
        <text>t_B_C02b1DE2b_B_Q02b2DE2_32</text>
      </name>
    </transition>
[...]
    <arc id="B_C02b1DE2b_t_B_C02b1DE2b" source="B_C02b1DE2b"
       target="t_B_C02b1DE2b">
      <inscription>
        <text>1</text>
      </inscription>
    </arc>
[...]
  </net>
</pnml>
```

Fig. 7. Example PNML Code

Our modeling approach does not generate any graphical information. Therefore a tool had to be found that is able to import PNML files, construct a graphical representation automatically, and analyse the P/T-net. We chose the ProM tool for this purpose. For more information on ProM, see [18] and [17]. ProM especially well supports the handling of large nets and arranges the net elements in a way that is very well readable for the human eye.

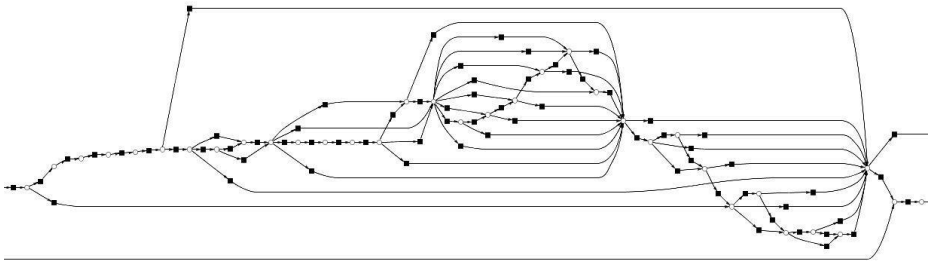

Fig. 8. BQ part as a P/T-net

In Figure 8 a small part of the complete BQ net is shown. As said before this is a simple model in the sense that the variables have been omitted. The figure is presented here just for illustration purposes. It serves to get an impression of the complexity of the overall BQ model. The complete model is way too big to be presented here. The layout of the example net has been done automatically by ProM.

Once available as a P/T-net in ProM, the build-in analysis means can be used. The PIAAC BQ has a single start item and a single end item. All items should be reachable and there may not be a dead end. The resulting BQ net therefore has to be a net with workflow properties. This is easily analyzable in ProM and gives good insight into the BQ definition. Doing so, we were able to find all the error types mentioned above with the big advantage of directly *seeing* the problems in the net graph. It now is way simpler to find fixes for the errors.

5 Advanced Net Models

In this section, we discuss experimental models that have not been used so far for the complete BQ. Nevertheless, as this is ongoing work, this will change soon.

To get deeper insight into the formal properties of a BQ, factoring in the variables and (routing) conditions is necessary. However, this may lead to way more complicated models, as we can see from a simple example. For this, we extend the example of Figure 6 to four possible answer categories on item `s1`, two answer categories on item `s2` and three conditional routings after `s3` relying on the variables `v1` and `v2`:

- `s4, if v1 = 1 and v2 = 1`
- `s5, if v1 = 2`
- `s6, if v1 > 2 or v2 = 2`

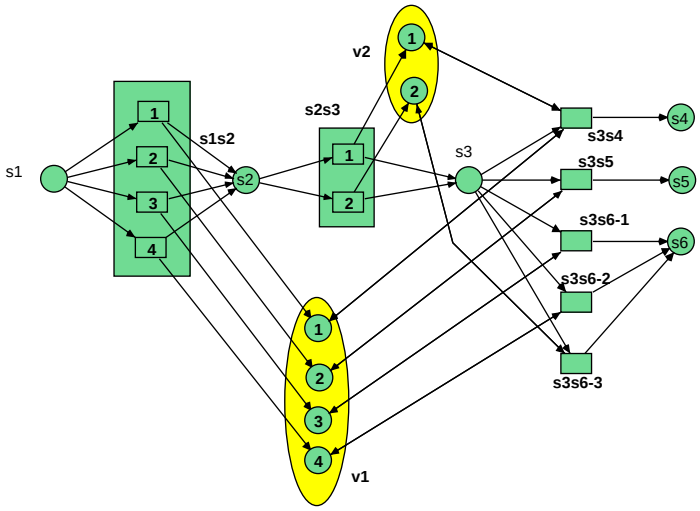

Fig. 9. Additional net elements for variables and conditions

Even this mild extension leads to a more complicated net model. Especially the last routing condition (leading to `s6`) is interesting, as it has to be unfolded to three transitions: two for the "greater-than" and and extra for the "or"-part. In real settings, this can easily grow to huge amounts of transition. As an example, in the German BQ there are routing conditions combining 5 variables, each enclosing up to 16 possible values, to route to more than 10 targets.

Another possibility is to model colored Petri nets instead of P/T-nets. Colored Petri nets directly support high-level data structures and variables. Nevertheless, they still allow for analysis, the necessary unfolding process is done inside the tool, for example the CPN Tools [4,10]. This option is under investigation.

In the moment, the PIAAC BQ is defined by means of writing XML code. This is tedious and error-prone work. The direct syntax can be checked relatively easily, but syntactical errors like missing routing targets are harder to detect. Semantic errors like dead ends even harder. Parts of these problems could be overcome by using a high-level Petri net formalism like Workflow Nets [9,15] as a means for rapid prototyping and/or adding small changes and corrections. Workflow Nets are directly executable, so that changes can be tried out easily. The graphical modeling permits typical errors mentioned above and gives a good overview on what one is doing. To support large BQ models, means for abstraction and rapid modeling are necessary. Workflow Nets offer such means.

Abstraction is possible in form of object tokens of the underlying reference net formalism [12,13,14] and by dynamic transition refinement [11]. Rapid modeling is facilitated by workflow patterns, a special form of net components - see [1] for an overview.

6 Conclusions and Outlook

We found a way of making use of well-known and well-understood formalisms and tools for a new domain. While some good results have already been achieved, several ways of extending the work are possible:

- The BQ analysis should be integrated into the normal BQ definition and release process. In the moment, it still requires manual work. It has been done completely only for the German version of the BQ.
- While the single steps of the analysis approach are rather straightforward, the combination still requires some manual work. This should be simplified to allow non-expert users to do the analysis on their own.
- The advanced models of Section 5 can be used directly for analysis of the BQ. This is still in an experimental state. We try to partition the BQ net into independent sub-nets to circumvent the net size explosion.
- In the moment, the BQ definition and especially the national adaptation process has long turn-around times. Countries request changes without the possibility to try them out beforehand. Using the rapid prototyping idea of Section 5 they could first try out the changes themselves and only request approval afterwards, once the changes are stable and working on the national level.

The examples and the analysis shown in this paper could partly be modeled using a sequential modeling formalism. However, Petri nets offer big advantages when it comes to non-local dependencies. For example, in the PIAAC BQ, some of the questions should only be asked a limited number of times in a country. This is straightforward to model in PN but maybe more difficult in other modeling formalisms.

The analysis of background questionnaires could benefit a lot from a sound formalization of what a BQ is. As mentioned early in the paper, no such definition exists so far. This question needs further research. Especially the similarity of BQs and workflows should be analyzed more deeply.

References

1. Lawrence Cabac. Net components: Concepts, tool, praxis. In Daniel Moldt, editor, *Petri Nets and Software Engineering, International Workshop, PNSE'09. Proceedings*, Technical Reports Université Paris 13, pages 17–33, 99, avenue Jean-Baptiste Clément, 93 430 Villetaneuse, June 2009. Université Paris 13.
2. Centre Research Public Henri Tudor (CRP-HT). http://www.tudor.lu. WWW.
3. Educational Testing Service (ETS). http://www.ets.org. WWW.

4. Computer Tool for Coloured Petri Nets (CPN Tools). http://wiki.daimi.au.dk/cpntools/cpntools.wiki. WWW.
5. Organization for Economic Co-Operation and Development (OECD). http://www.oecd.org. WWW.
6. Programme for International Student Assessment (PISA). http://www.pisa.oecd.org. WWW.
7. Programme for the International Assessment of Adult Competencies (PIAAC). www.oecd.org/els/employment/piaac. WWW.
8. L.M. Hillah, E. Kindler, F. Kordon, L. Petrucci, and N. Trèves. A primer on the petri net markup language and iso/iec 15909-2. *Petri Net Newsletter*, 2010.
9. Thomas Jacob, Olaf Kummer, Daniel Moldt, and Ulrich Ultes-Nitsche. Implementation of workflow systems using reference nets – security and operability aspects. In Kurt Jensen, editor, *Fourth Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, Ny Munkegade, Bldg. 540, DK-8000 Aarhus C, Denmark, August 2002. University of Aarhus, Department of Computer Science. DAIMI PB: Aarhus, Denmark, August 28–30, number 560.
10. Kurt Jensen, Lars Michael Kristensen, and Lisa Wells. Coloured petri nets and cpn tools for modelling and validation of concurrent systems. *International Journal on Software Tools for Technology Transfer (STTT)*, Volume 9(3):213–254, 2007.
11. Michael Köhler and Heiko Rölke. Dynamic transition refinement. *Electronic Notes in Theoretical Computer Science*, 175:119–134, June 2007.
12. Olaf Kummer. *Referenznetze*. Logos Verlag, Berlin, 2002.
13. Olaf Kummer, Frank Wienberg, Michael Duvigneau, and Lawrence Cabac. Renew – the Reference Net Workshop. Available at: http://www.renew.de/, August 2009. Release 2.2.
14. Olaf Kummer, Frank Wienberg, Michael Duvigneau, and Lawrence Cabac. *Renew – User Guide*. University of Hamburg, Faculty of Informatics, Theoretical Foundations Group, Hamburg, release 2.2 edition, August 2009. Available at: http://www.renew.de/.
15. Daniel Moldt and Heiko Rölke. Pattern based workflow design using reference nets. In Wil van der Aalst, Arthur ter Hofstede, and Mathias Weske, editors, *Proceedings of International Conference on Business Process Management, Eindhoven, NL*, volume 2678, pages 246–260, 2003.
16. Petri Net Markup Language (PNML). http://www.pnml.org/. WWW.
17. Process Mining Toolkit (ProM). http://prom.win.tue.nl/tools/prom/. WWW.
18. W.M.P. van der Aalst, B.F. van Dongen, C. Günther, A. Rozinat, H. M. W. Verbeek, and A. J. M. M. Weijters. Prom: The process mining toolkit. In *Proceedings of the BPM 2009 Demonstration Track, Volume 489 of CEUR-WS.org, Ulm, Germany*, 2009.
19. XSL Transformations (XSLT). http://www.w3.org/tr/xslt. WWW.