2 RESEARCH METHODOLOGY AND STRUCTURE The research presented in this paper follows a design science research approach (DSR). This approach was chosen because of the proximity of the investigated research questions to the practical problems and the intention to develop artifacts that have a high contribution and relevance for the application domain. €Osterle et al. [18] suggest following a research pro- cess that consists of the four phases analysis, design, evalua- tion and diffusion. The structure of this article follows these phases. The analysis phase is represented in the subsequent sections that illustrate the state-of-the-art of related scientific work and discuss the requirements for the development of the presented mining algorithm. The design of the mining algorithm as the core contribution of the paper is presented in section five. Scholars like Hevner et al. [19] stress the importance of research rigor in DSR. We have therefore included an evaluation section that illustrates the results that have been achieved by applying the presented algorithms to extensive real world data. The paper closes with a discussion of the research results, relevant limitations, outlook to future research and a brief summary. We used different research methods with varying para- digmatic orientation during the research process to reduce the risk of paradigmatic bias and to investigate the research problem from different angles [20]. The results presented in this paper are part of larger research effort that has been carried out by several researchers over the recent years. The development of the presented algorithm bases on the application domain requirements that were identified in empirical studies using expert interviews and surveys [21]. The main research methods that were used to develop the presented algorithm were method engineering [22] and prototyping [23]. Method engineering is an approach for assembling novel methods based on already existing or newly developed method fragments. Research outcomes from prior research have been incorporated as input for the presented research to develop a novel algorithm. Relevant research results that have been considered in the course of method engineering include solutions for the calculation of instance graphs, projections, aggregation graphs [24], and causal matrices [14] as well as previously published research results from the authors concerning the integration of the data perspective into process models [25], case matching [17], complexity reduction of mined models [26], and data dependent control flow inference [27]. The designed algo- rithm was instantiated in a software prototype to enable WERNER AND GEHRKE: MULTILEVEL PROCESS MINING FOR FINANCIAL AUDITS 821 laboratory simulation experiments. Data sets from industry partners were used as input for the prototype to evaluate the proper functioning of the implemented algorithm and to analyze themining results.

3 RELATED WORK The research presented in this paper deals with the question of how process mining can be used as a BI approach for financial audits. Process mining is a research domain that has matured over the past decades. It would go beyond the scope of this paper to provide a complete overview of process min- ing research. Instead we refer to literature that provides a good overview. Tiwari et al. [28] provide a survey of the state- of-the-art and future trends in process mining until the year 2008. Basic and advanced process mining concepts have been comprehensively summarized by van der Aalst [3]. He pro- vides an extensive collection of main research results that have been achieved in recent years and presents an overview of contemporary opportunities and challenges [29]. Compliance checking is a process mining application area that is of particular interest to the research at hand. Compli- ance can be defined as the adherence to internal or external rules. The main objective of financial audits is to ensure that companies adhere to accounting standards, laws and regula- tions. Debreceny and Gray [30] suggest the application of data mining methods for analyzing journal entries and pro- vide an extensive case study. Becker et al. [31] have researched the applicability of model-based business process compliance-checking approaches and developed a classifica- tion framework. They provide a literature review and distin- guish between forward- and backward-compliance checking approaches. Process mining for financial audits is a back- ward-compliance checking approach because its objective is the disclosure of compliance violations after they have occurred and been recorded in the event log. The application of process mining as a compliance checking approach has already been addressed by different scholars. Alles et al. [32] propose the application of process mining in accounting information systems. Jans et al. highlight opportunities and challenges for using process mining as an audit tool [33] and provide interesting case studies [11], [34]. They focus on the control flow and organizational perspective. An important difference between internal and external audits is the rele- vance of the data perspective. M€uller-Wickop et al. [21] con- ducted empirical research which shows that internal and external auditors do not necessarily share the same perspec- tive on the importance of different application domain con- structs. We will discuss the requirements needed to use process mining in financial audits in greater detail in the sub- sequent section. But it is important to mention that the inclu- sion of the data perspective is a key requirement which has not been considered by prior research. The lion’s share of pro- cess mining research deals with the discovery of the control flow whereas the integration of the data perspective in pro- cess mining has generally not been investigated extensively in the academic community yet, apart from very few scien- tific publications. This observation is supported by Stocker [35] and de Leoni and van der Aalst [15]. De Leoni and van der Aalst use the data flow perspective to discover rules that explain why instances of the same process follow different execution paths. They introduce variables as net components for the extension of Petri Nets (DPN-nets). Trcka et al. [36] follow a different research question to discover data flow errors but apply a similar approach by using extended workflow nets (WFD-nets). Accorsi and Wonnemann [37] choose a different approach to identify information leaks in process models. They include data objects as colored tokens in Colored Petri Nets (CPN).We follow a similar approach and use CPN to include the data perspective by modeling data objects as colored tokens [25]. This allows us to present the control flow and data flowperspective in a singlemodel. A fundamental challenge for process mining is the balanc- ing between competing quality criteria [9]. Process mining is generally used to reduce complexity by visual representation and abstraction. A process model represents a set of process executions which are called process instances.1 Multiple exe- cutions of a business process commonly do not occur in exactly the samemanner. Variety in the execution leads to dif- fering process instances. Every organization needs flexibility to adapt business activities to changing customer demands and market influences. A certain degree of deviation is there- fore neither surprising nor damaging. But variance in the course of execution means that it can get impossible to create a model that unambiguously describes the represented pro- cess. This leads to the phenomenon of miss-fitting process models. A model is under-fitting if it allows execution paths in the process model that are not represented in the event log and over-fitting if they do not allow for any additional behav- ior that is not included in the event log.2 Rozinat et al. provide a framework for the evaluation of process mining algorithms. They identify four quality criteria for the evaluation: fitness, precision, generalization and structure [38]. Fitness indicates if a model is able to represent all cases in the event log. Preci- sion is the complementary criterion. It indicates if a process model does not allow additional behavior that was not observed in the log. Generalization addresses the capability of a model to express more behavior than recorded in the log. It is generally desirable to create a process model that shows an adequate degree of generalization. Structure refers to the graphical representation of a business process and depends on the graphical components of the target language. Other scholars use the closely related criterion of simplicity instead of structure [9]. The following section will show that contrary to many other application domains financial audits require perfectly fitting and as precise process models as possible. De Medeiros et al. suggest the clustering of cases that exhibit sim- ilar traces in the event log.3 Process models are then gener- ated for each cluster [39]. This approach prevents over- generalization and is very useful because it does not require using a specific mining algorithm. A similar trace clustering 1. The term “process instance” and “case” are used ambiguously among scholars. We refer to “process instances” as real world execu- tions of business processes whereas “cases” represent a record of a pro- cess instance in an event log. A case is therefore a purposeful abstraction of a process instance represented as a data record. 2. The concepts of under- and over-fitting process models refer to the definition established in [3]. Under- and over-fitting refer to charac- teristics of mined process models whereas noise and incompleteness refer to characteristics of the event log. 3. A trace is the recorded sequence of executed activities in a process instance. Every case has a specific trace but different cases can exhibit identical traces. 822 IEEE TRANSACTIONS ON SERVICES COMPUTING, VOL. 8, NO. 6, NOVEMBER/DECEMBER 2015 approach is used by Song et al. [40]. Van der Aalst et al. [41] provide a powerful mining algorithm for balancing between over- and under-fitting by using the theory of regions to cre- ate Petri Nets from transition systems. The approaches by de Medeiros et al., Song et al. and van der Aalst et al. are very valuable but they do not consider the data perspective and the idiosyncratic structure of journal entries that form the basis for the event log in financial audits. 4 REQUIREMENTS This section describes the requirements for using process mining algorithms in financial audits. A business process consists of activities. Information systems that support or automate the execution of activities create journal entries that are posted to the relevant financial accounts. Fig. 1 illus- trates this relationship and provides an example of a simple purchase process that consists of four activities. Three of the four illustrated activities (B, C, D) create journal entries on different accounts. The process starts with the ordering of goods (A). This activity does not create any journal entry on a financial account because the order- ing itself has no impact on the balance sheet or income state- ment. This changes when the ordered goods are received. The corresponding activity (B) creates two journal entries, a debit posting on the RawMaterials account and a credit post- ing on the Goods Received / Invoices Received (GR/IR) account. The next activity (C) clears the open entry posted by activity (B) with a debit posting on the IR/GR account and a corre- sponding credit posting on the Trade Payables account. The process finally terminates with the payment of the received invoice (D) that creates a debit posting on the Trade Payables and a credit posting on the Bank Account. All the postings are stored in the information system that is used to operate the business process. M€uller-Wickop et al. [21], [42] conducted empirical investi- gations using expert interviews and surveys to identify key concepts and information requirements for process audits. They show that the process flow is a central concept and highly relevant froman audit perspective in practice. To audit a process it is necessary to understand how it is structured and which control activities are included that safeguard the correct processing of transactions. This requirement can be satisfied by modeling the control flow perspective. Two fur- ther important concepts for external auditors are financial statements andmateriality4. Knowledge about the interaction among activities alone is not sufficient. For the auditor it is necessary to understand how the activities relate to the finan- cial accounts as illustrated in Fig. 1. Only those business transactions are inspected in a finan- cial audit that can have a material effect on the financial statements. It is therefore necessary to receive information on the value flow that is created by the audited business process to decide if it needs to be audited from a materiality perspective or if it can be neglected. Another critical requirement in financial audits is the preservation of the audit trail. The audit trail is a fundamental concept in financial accounting. It is a path in an information system that allows tracing a transaction from the point of origin to the final output. It is used to verify the accuracy and validity of journal entries [44]. Translating this requirement into the context of process mining implies that a mining algorithm may not alter the original data during the mining process. A suitable mining algorithm must fur- ther be able to present the unchanged source data to the auditor for investigation purposes. But on the other hand the mining algorithm should also be able to present informa- tion at an adequate abstraction level to provide an overview of the control and data flow as discussed before. If process mining is used in financial audits the generated models are used to discover incompliant behavior. The provided process models should therefore be as precise and fitting as possible. If the produced process models are over- fitting certain behavior recorded in the event log is not rep- resented in the process model. The auditor would therefore assume that no incompliant behavior has occurred. But in reality it is just not represented in the miss-fitting process model which would eventually lead to false positive audit results. If the process model is too general, process behavior is illustrated that actually did not occur. This would lead to false negative audit results and unnecessary investigations by the auditor. The quality criteria identified by Rozinat et al. [38] are useful to express the requirement of accurate process models in terms that are applicable for the process mining research domain. Simplicity of a process model is a preferred characteristic but it is not a key requirement in financial audits. Auditors currently spent weeks trying to understand a business process with the use of traditional audit procedures and by reviewing hundreds of documents. It is therefore acceptable if process models are complex and in extreme cases only comprehensible to experts. Neverthe- less a mining algorithm should be able to deliver process models as simple as possible. Generalization should be min- imized and precision maximized to prevent false negative compliance testing results. Process models should be per- fectly fitting to possibly represent all recorded behavior and to prevent that incompliant behavior that actually occurred remains undetected and therefore reduce the audit effec- tiveness. Different metrics can be used to measure fitness (completeness [45], PFcomplete [14], fitness (f) [46], parsing mea- sure (PM) and continuous parsing measure (CPM) [12]. The metrics completeness and PM calculate the percentage of traces in the log that can be replayed by the model. The other three metrics consider both traces and tasks in a model. A process model has a perfect fitness if the metrics have a value of one. The precision dimension can also be measured by using different metrics (soundness [45], behav- ioral appropriateness [46], and behavioral precision [14]). A per- fect precision is reached if the relevant metrics also take on the value of one. Fig. 1. Simple purchase process. 4. Materiality is defined in ISA 320: “Misstatements, including omis- sions, are considered to be material if they, individually or in the aggre- gate, could reasonably be expected to influence the economic decisions of users taken on the basis of the financial statements.” [43]. WERNER AND GEHRKE: MULTILEVEL PROCESS MINING FOR FINANCIAL AUDITS 823 5 MULTILEVEL PROCESS MINING 5.1 Mining Algorithm The requirements discussed in the previous section are par- tially conflictive. Listing 1 shows an abstract and simplified version of the mining algorithm that was developed based on the identified requirements. A fundamental aspect of the presented solution is to satisfy different requirements at dif- ferent abstraction levels. The algorithm is called Multi- Level-Process-Mining algorithm (MLPM) due to its main feature of being able to produce process models at different abstraction levels. The MLPM first matches event data to cases (line 1 to 11), then creates instance graphs (line 12 to 21), calculates pro- cess instance models (line 22 to 34) and aggregates instance models to process models (line 35 to 45). Each section is dis- cussed in detail in the following subsections. 5.2 Case Mining and Event Log Structure The mining algorithm uses recorded transaction data from ERP systems as the data source for the mining. This kind of data only has a medium maturity level from a process min- ing perspective because ERP systems like SAP or JD Edwards do not use a systematic approach to link and store process relevant data [9]. The event log data in such systems is stored in different database tables and needs to be com- posed in a meaningful manner before it can be used as input for process mining algorithms. Data related to financially rel- evant business transactions exhibits specific characteristics that can be used for processmining purposes [25]. The execu- tion of financially relevant transactions in an ERP system cre- ates specific data records. Every execution of an activity creates a posting document. A posting document is a data record that represents a journal entry. Each document LISTING 1 Multilevel Process Mining Algorithm 1. Mine Cases 2. D set of all posting document numbers 3. J set of all journal entry item numbers 4. ID set of all case IDs 5. Di ¼; initially empty set of document numbers belonging to case i 2 ID 6. Ji ¼; initially empty set of journal entry item numbers for case i 2 ID 7. While D 6¼ ; 8. Remove d 2 D from D and insert d into Di 9. Insert all j 2 J posted by d into Ji and remove j from J 10. Insert all d 2 D that cleared j 2 Ji into Di and remove d from D 11. Repeat 9. and 10. for all d 2 Di and j 2 Ji 12. Reconstruct Instance Graphs 13. IG ¼ ; initially empty set of instances graphs 14. IGi instance graph (Ni;Ei;Li;li) for case i with the set of nodes 15. Ni 6¼ ;, Ei Ni Ni is the set of arcs, Li the set of task labels 16. and li: Ni ! Li is a labeling function mapping nodes onto Li 17. For all i 2 ID 18. Create n 2 Ni for each d 2 Di with liðnÞ ¼ transaction code of d 19. Create eðnj; nkÞ2 Ei for each dj and dk 2 Di if dk cleared an item 20. j 2 Ji that was posted by dj 21. Insert IGi into IG 22. Reconstruct Instance Models 23. IM ¼ ; initially empty set of instances models 24. IMi instance model (Ti;Pi;Ai, P i, ViCiGi;Ei;Ii)for case i2ID 25. For all i 2 ID 26. Set Ti ¼ Ni 27. For each eðnj; nkÞ2 Ei create p 2 Pi, aðtj; pÞ, aðp; tkÞ 28. For each d 2 Di 29. Create p 2 Pi for each j 2 Ji that was posted by d 30. Create aðt; pÞ 2 Ai for each j 2 Ji that was posted by d and 31. aðt; pÞ, aðp; tÞ2 Ai for each j 2 Ji that was cleared by d 32. Aggregate all places pk and pj 2 Pi if CiðpkÞ ¼ CiðpjÞ 33. Aggregate all transitions tk and tj 2 Ti if liðtkÞ ¼ liðtjÞ 34. Insert IMi into IM 35. Mine Process Models 36. PM ¼ ; initially empty set of process models 37. Compute causal matrix CM(IMi) for all i 2 ID 38. While IM 6¼ ; 39. Remove IMj from IM and insert IMj into PM 40. For each IMk 2 IM 41. If CMðIMjÞ ¼ CMðIMkÞ 42. Merge IMj and IMk with Tjk ¼ Tj [Tk, Pjk ¼ Pj [Pk, Ajk ¼ Aj [Ak, Sjk ¼ P j [ P k 43. Aggregate all places pl and pm 2 Pjk if CjkðplÞ ¼ CjkðpmÞ 44. Aggregate all tl and tm 2 Tjk if ljkðtlÞ ¼ ljkðtmÞ 45. Remove IMk from IM 824 IEEE TRANSACTIONS ON SERVICES COMPUTING, VOL. 8, NO. 6, NOVEMBER/DECEMBER 2015 contains at least two journal entry items. This connection is illustrated by the ‘contains’ relationship in the ER model pre- sented in Fig. 2. Journal entries that follow an open-item- accounting principle are linked to each other if they belong to the same process instance. If a process instance is termi- nated each open journal entry item has been cleared by another posting document.5 This dependency is modeled in Fig. 2 via the ‘is cleared’ relationship. It is possible to exploit this data chain for each process instance. The procedure is illustrated in line 1 to 11 in Listing 1. The algorithmmaps event log entries to cases. It starts with a document number andmines all related journal entry items by using the ‘contains’ relationship illustrated in Fig. 2. It then searches for all document numbers that have cleared these items by using the ‘is cleared’ relationship. All posted items for every clearing document are then searched in further iter- ations. The loop terminates when all documents and items that belong to the same process instance have been found. The loop itself is repeated until all case IDs have been mined. The outcome is an event log [ni¼1 Di; Jif g with Di ¼ Document Nrd1; . . . ; Document Nrdmgf 8 posting documents d1 . . . dm match¼) i and Ji ¼ ItemNrj1; . . . ; ItemNrjmg8items j1 . . . jm match¼) i. The event log also includes the data attributes associated to each posting document and journal entry item that are listed as entity attributes in Fig. 2. In contrast to tradi- tional event logs the events belonging to a single case do not follow a strict linear order. The causal dependencies between the events have to be determined in a separate step. 5.3 Instance Graphs and Abstraction Levels The application of process mining for financial audits requires algorithms that enable the investigation of a finan- cially relevant transaction from its point of origin to the post- ing on the financial accounts. On the other hand an auditor should be able to receive an overview to get an all-encom- passing understanding of the process structure and its effect on the financial accounts. Both requirements can be met by using different abstraction levels. Scholars commonly distin- guish between four levels of horizontal abstraction in busi- ness process management [47]. The instance level represents tangible entities that are involved in business processes which include executed activities, resources, concrete data values etc. A set of similar business processes are represented as business process models on the model level. A model is expressed using constructs of a meta-model. These can them- selves again be defined at themeta-meta-model level. Process mining algorithms commonly operate on the model level. They generate process models that are abstractions of the individual process instances they represent.6 Fig. 3 illustrates how the different abstraction levels that were used to develop the MLPM relate to each other and what kind of models the algorithm generates. Process instance graphs reside on the lowest level of abstraction. They are graphical representa- tions of the source data from executed and recorded process instances. Fig. 3a shows two instance graphs. They represent two executions of the purchase process illustrated in Fig. 1. Each rectangle represents the execution of an activity and the arcs between two activities denote the causal relationships between executed activities. The numbers indicate how often an activity was executed and how often the path from one activity to another was chosen. Instance A in Fig. 3a shows the distinctive but typical structure of process instances recorded in ERP systems because it has two branches mean- ing that two invoices (C) for two different received goods (B) were paid by the same payment (D). The process instance model is an abstraction of the instance graph (Fig. 3b). Execu- tions of identical activities are aggregated into activitymodels but an instance model still represents just a single execution of a business process. A process model is an abstraction of a set of similar process instances (Fig. 3c). The presented pro- cess instance and process models also provide information on the involved financial accounts and hence incorporate the data flow perspective (compare Sections 5.4 and 6).7 Fig. 2. Entity-relationship-model for accounting data structure. Fig. 3. Abstraction levels in the context of process mining7. 5. This principle is also illustrated in Fig. 1. Activity (B) creates an open credit posting on the IR/GR account that is cleared by a debit posting on the same account from (C). The open item posted on the Trade Payables account from (C) is cleared by (D). The process terminates with a posting on the Bank Account. This account is not enabled for open-item-account- ing and hence all posted items are cleared after the execution of (D). 6. An exception is the multi-phase process mining approach that was developed to generate event-driven-process-chains [24]. It operates on the instance and model level. A similar approach is used for the design of the MLPM. 7. The models in Fig. 3a represent simple directed graphs without logical operators. It is not necessary to model choice at the instance graph level because all decisions have already been made and all rela- tionships in Fig. 3a have the semantic of an AND split or join. The mod- els presented in Figs. 3b and 3c represent abstractions of the lower level models. The relationships on these level can generally represent AND, OR and XOR splits and joins. An algorithm to transform instance graphs into EPC or Petri Nets is described in [24]. WERNER AND GEHRKE: MULTILEVEL PROCESS MINING FOR FINANCIAL AUDITS 825 The process model is important for the auditor to get an overview about the structure of the process, its relationships to the financial accounts and to assess the overall materiality of a business process. Process instance models and process instance graphs are useful for following the audit trail and for inspecting individual process instances with respect to involved activities, users, data values etc. The second part of the mining algorithm ranging from line 12 to 21 creates instance graphs. The algorithm first cre- ates a node for every document number labeled with the transaction code that was used to create the posting docu- ment in the ERP system (line 18). Different nodes in the instance graph can carry the same label (compare Fig. 2a) if they were created by the same transaction of the ERP system. The algorithm then infers the causal dependency between activities (line 19 and 20). Traditional mining algorithms rely on the time stamp of events to infer the control flow. This is not suitable for the event log created by the MLPM algo- rithm. The same posting document can clear journal entry items belonging to different other posting documents lead- ing to parallelism in the event log. Using the temporal order- ing of events can lead to intertwining in parallel branches resulting inmodels that are of little use to the user.8 The algo- rithm uses instead a data dependent approach for determin- ing the control flow. Sun and Zhao [49] introduced an approach to derive the control flow by modeling the data flow. The data dependencies of an activity v can be expressed as dvðIdv; OvÞ, where Iv is the input for v, andOv the output. d denotes the type of data dependency. The algorithm analyses in lines 19 and 20 how activities relate to each other based on their data relationships. It checks if activity A has cleared ItemB posted by activity B. If the condition is true a control arc from B to A is inserted. A can only have occurred if B took place before. Otherwise there would have been no ItemB that could have been cleared by A. This is equivalent to a mandatory dependency between A and B denoted as B!mA because of OB\ IuA 6¼ ;with ItemB 2 OB^ItemB2IuA. The results of these operations are instance graphs in form of directed graphs equivalent to the graph shown in Fig. 3a. 5.4 Instance Models and Colored Petri Nets The third section of the mining algorithm reconstructs pro- cess instance models as CPN. CPN are suitable for the modeling of business processes and offer a formal as well as graphical notation that can even be understood by non- experts [50]. They provide a sound mathematical founda- tion for the simulation and verification of Petri Net models [51]. The majority of process mining algorithms rely on low- level Petri Nets. An exception is the approach used by Accorsi and Wonnemann [37]. They use CPN and model data objects as colored tokens. We use a similar approach to generate process models that model the control flow and data flow perspective simultaneously in a single model. A Colored Petri Net is formally expressed by the tuple CPN ¼ ðT; P;A;S; V; C;G;E; IÞ [51, p. 87], with: 1) T is a finite set of transitions. 2) P is a finite set of places. 3) A 2 P T [T P is a set of directed arcs. 4) S is a set of non-empty color sets. 5) V is a finite set of typed variables such that Type½v 2 S for all variables v 2 V . 6) C : P ! S is a color set function that assigns a color set to each place. 7) G : T ! EXPRV is a guard function that assigns a guard to each transition t such that Type G tð Þ½ ¼ boolean. 8) E : A ! EXPRV is an arc expression function that assigns an arc expression to each arc a such that Type E að Þ½ ¼ CðpÞMS; where p is the place connected to the arc a. 9) I : P ! EXPR; is an initialization function that assigns an initialization expression to each place p such that Type I pð Þ½ ¼ CðpÞMS . We integrate the data perspective by modeling colored places in the CPN that represent financial accounts [25]. An instance graph produced by section Reconstruct Instance Graphs is first transformed into a CPN. The nodes of the instance graph become the transitions of the CPN (line 26).9 We distinguish between two types of places. Control places model the control flow and account places model the data flow. Each arc from the instance graph is transformed into a combination of a control place and two connecting control arcs in the instance model. A source place psource is inserted with arcs aj psource; tj 8 tj 2 T ^ tj ¼ ; and a sink place psink is with arcs ak tk; psinkð Þ8tk 2 T ^ tk ¼ ;. The data perspective is integrated and visualized by creating account places for each journal entry item in the event log (line 29). The color set function C assigns different color sets to places depend- ing onwhether they belong to the group of control or account places.10 The set of color sets S includes the color sets for all possible journal entry values, account numbers, account types, credit or debit indicators and execution numbers. Account places are connected to related transitions (line 30 and 31). A simple arc a t; pð Þ is inserted from transition t to the place p if the transition posted a journal entry item on the represented financial account. They are referred to as posting arcs. Two arcs a t; pð Þ ^ aðp; tÞ are inserted if the transition has cleared an item on the respective account. These arcs have the semantic of testing arcs because they do not consume the tokens on the connected places. Dou- ble-headed arcs are used for visualization as a syntactical abbreviation for two arcs aðt; pÞ ^ aðp; tÞ. They are called clearing arcs. The arc inscriptions are modeled as constants (V ¼ fg). The arc expression function E assigns to each posting and clearing arc a set of constants that denote the posted or cleared value, the account type, account number and an indicator whether it is a credit or debit posting. Each control arc is assigned a number that indicates how often this represented control path was chosen in the instance.10 The source place is initialized by the 8. A discussion of this aspect is beyond the scope of this paper but it is illustrated in detail in [27]. The event logs can be converted into linear event logs [48]. But this transformation is accompanied with undesired side-effects like the duplication of events in the log, and it is not able to deal with the data perspective. 9. Guards are not needed and the guard function G is therefore defined as GðtÞ ¼ true for all t 2 T . 10. A detailed description of the color set function C and arc expres- sion function is provided in [25]. 826 IEEE TRANSACTIONS ON SERVICES COMPUTING, VOL. 8, NO. 6, NOVEMBER/DECEMBER 2015 initialization function I. The initialization expression for psource generates n tokens in the initial marking M0ðpÞ, one for each connected start transition. After having integrated the data perspective by creating a CPN it is necessary to aggregate model components to cre- ate instance models. The first step is aggregating places rep- resenting financial accounts. Two places pk and pj can be aggregated if they represent the same account. This is the case if they carry the same color CðpkÞ ¼ CðpjÞ (line 32). The relationships between the places and connected transitions need to be maintained in respect to the type and inscription of each arc.11 The second step aggregates the transitions (line 33). The used method for aggregating transitions is based on the algorithm described by van Dongen and van der Aalst [24]. Transitions are aggregated if they carry the same label lðtjÞ ¼ lðtkÞ. The result of the third section of the mining algorithm is a set of instance process models repre- sented as CPN (compare Fig. 3b). 5.5 Process Models and Clustering The final section of the mining algorithm produces process models (lines 35 to 45). A key requirement for the application of process mining in the context of financial audits is the cre- ation of perfectly fitting and highly precise process models. Some researchers have developed methods to create precise and fitting process models by clustering traces in the event log [39], [40]. Their approaches are not directly applicable because they do not take into account the data perspective and because they are not suitable for the given event log structure. But clustering is an approach that is also used for the creation of process models in the final section of the pre- sented mining algorithm. But instead of clustering traces we cluster and aggregate process instance models. We use causal matrices to identify isomorph process instances. Causal matrices are key components for heuristic [12] and genetic mining algorithms [52, p. 6]. A causal matrix describes the causal relation between activities in a model. An entry in a causal matrix in column j and row k designates that a causal relation exists between the activities j and k. The causal matrices of two instance models are identical if they show exactly the same behavior. The algorithm com- putes the causal matrices for all instance models (line 37) that were created by the previous operations. It then searches for identical causal matrices (line 41) and aggregates two models if they have the same causal matrix (lines 42 to 44). The aggregation procedures are identical to the procedures used at the instance model level (line 32 and 33). The aggre- gation procedure is repeated for all instance models until the set of instancemodels is empty (line 38). Aggregating process instance models that exhibit identi- cal control flow patterns does not mean that the resulting process model is identical to the source process instance models. The data values in each process instance model are unique and are preserved during the aggregation process. A process model therefore shows the aggregated data values and data flow of all represented process instances. The final outcome of the mining algorithm is a set of process models where each model represents process instances with the same control flow pattern. Each process model provides the aggregate view of the data flow and relationship of activities with the financial accounts (compare Figs. 3c and 4). 6 MINING RESULTS AND EVALUATION The aim of the conducted evaluation is to test whether the designed algorithm is able to produce the estimated results and to get insights into the characteristics of mined process models from real world data. The estimated results are models representing information at different abstraction levels that visualize the control and the data flow, the ability to learn models from unlabeled event logs from ERP sys- tems, perfectly fitting models to prevent false positive audits results, and as precise process models as possible to avoid false negative audit results. An artificial evaluation is appropriate for the research at hand because it is not necessary to observe the behavior in organizational contexts at this stage. We have chosen an artificial laboratory experiment as one of the evaluation methods suggested for this setting by Venable et al. [53]. Such an evaluation requires the existence of instantiated artifacts that can be used for the experiment. The different sections of the mining algorithm were therefore imple- mented in a software artifact in iterative cycles. The experi- ment itself was divided into the phases data extraction, mining and results analysis. We used a separate software module for extracting relevant data from ERP systems12 that can be adjusted to different source systems. We checked the data for first and second order defects [6, pp. 29-32]. Three data sets were used for the evaluation. We extracted data from the productive SAP systems of three companies operating in the retail, manufacturing and media industry. The MLPM was used to mine all process instance graphs, process instance models and process mod- els for the three data sets. The data set characteristics and mining results are summarized in Table 1. The models were inspected on a sample basis by observa- tion and comparison with the original event log data. The software yEd—Graph Editor [54] provides powerful auto- matic layout functionality and is free for use for noncommer- cial purposes. It uses the GXML and GraphML formats as input and was used in the experiment to graphically repre- sent the process models. The models were further tested for soundness (proper completion, option to complete, absence of dead transitions, and safeness for all places except account places) by using the CPN simulation tool Renew [55]. Fig. 4 shows an example of a mined process model from data set 1. It is modeled as a CPN as specified in Section 5.4. The transitions illustrated as rectangles represent the activi- ties. The mined model represents a purchasing process that was executed by 8 identical process instances. It contains the activities Post Goods Receipt, Enter Incoming Invoice, Clear Account, Payment and Post with Clearing that were executed in a sequential order. Control places are rendered with a bold black border. Simple arrows between transitions and control 11. A complete description of this procedure is beyond the scope of this paper. It is described in [26]. 12. The introduction of this module is beyond the scope of this paper. The relevant data is stored in the tables BKPF and BSEG in the case of SAP ERP systems. WERNER AND GEHRKE: MULTILEVEL PROCESS MINING FOR FINANCIAL AUDITS 827 places model the control flow and sequence of activities. The inscriptions for these arcs show how often the path was cho- sen in the represented instances.13 Account places have a solid or dashed border. The type and color of the border line indi- cate if the represented account is a balance sheet or profit and loss account and if it represents the debit or credit side.14 Account places and transitions are connected by dotted arcs. A simple dotted arrow is a posting arc. Double-headed dotted arcs are clearing arcs. Posting and clearing arcs visualize the data flow in the process model. The Post Goods Receipt activity for example creates a tokenwith the value of 593,846.02 repre- senting a journal entry item posting on the raw materials account 0001421100. Another token is created on the account 0002810200. This is cleared by the subsequent activity of Clear Account but without consuming the token on the respective account. The modeled CPNmimics the behavior of the repre- sented process. Transitions create colored tokens on the account places representing the posted journal entries. The control places define the control flow in themodel. The model is perfectly fitting and precise because it can replay all process instances and does not allow any addi- tional behavior. Fitness and precision can be measured using different metrics such as fitness ðfÞ and behavioral appropriateness ðaBÞ [46]. These measures can be calculated by replaying cases from the event log and by counting if tokens are missing in the CPN in order to execute the simulation or remain unconsumed in the model after the simulation is terminated. These matrices are not directly applicable to the used CPN. Colored data tokens remain in the model by definition even after all transitions have fired. The used events in the event log do not follow a strict linear order for each case. It is therefore unclear which sequence of events should be used for replaying a case. The traces can be transformed into strict linear sequences [48] but these sequences would not fit to the mined model anymore. The possible transformation into linearized traces would intro- duce choice at the instance level that did not occur in reality. The model illustrated in Fig. 4 does not show any choice. The paths p1 ¼ B ! D and p2 ¼ B ! C ! D are parallel and not optional paths. The trace A ! B ! D ! E for example, which would be an output of the transformation, cannot be replayed by the model presented in Fig. 4.15 Calculating token based measures like ðfÞ and ðaBÞ for the produced process model would require many artificial adjustments which are not reflected by the actual source data. We alternatively use metrics that are suitable to take into account possible parallelism at the instance level and that directly compare the execution paths in the different models. We use the percentage of the control flow paths from psource ! psink that are present in the instance graphs to those in the process model for measuring the fitness denoted as fPath. And we compute the percentage of the control flow paths in the processmodel that are not present in the instance graphs formeasuring the precision denoted as pPath with: fPath ¼ pjp 2 PPM^ 2 PIGf gj j PIGj j ; pPath ¼ PPMj j pjp 2 PPM ^ =2 PIGf gj j PPMj j ; PPM ¼ p1; . . . ; pnf g is the set of distinct control flow paths in the process model PM and PIG ¼ p1; . . . ; pmf g is the set of all distinct control flow paths from all instance graphs IG that belong to PM. TABLE 1 Evaluation Data Sets Set 1 Set 2 Set 3 Industry Manufacturing Media Retail Posting Documents 1,764,773 156,604 92,487 Journal Entry Items 7,395,434 559,506 222,901 Process Instances 1,035,805 18,975 40,634 Process Models 841 516 307 Fig. 4. Example of a mined process model13. 13. The arc inscriptions for posting and clearing arcs only display the assigned constant for the posted or cleared value. The inscriptions for the account type, account number and credit or debit indicator are omitted to improve readability. The same is the case for the inscriptions of the connected account places that only show the account number. 14. Solid line ¼ balance sheet account, dashed line ¼ profit and loss ac-count, black line ¼ debit side of an account, gray line ¼ credit side of an account. 15. The model itself can be transformed in such a way that it is able to replay all linearized traces. But this would result in a much more complex models and a negative effect on the model precision. 828 IEEE TRANSACTIONS ON SERVICES COMPUTING, VOL. 8, NO. 6, NOVEMBER/DECEMBER 2015 Van Dongen and van der Aalst [24] proof that the used aggregation procedures are path preserving. This means that all control flow paths are also represented in the process model with fPath¼ 1. We calculated pPath for the mined mod- els from data set 1 to 3. The results are listed in Table 2.16 The average value for this measure over all three data sets is 0.81. It does not equal 1 due to the potential parallelism of branches on the process instance graph level and differences in the exe- cution paths of parallel branches.17 A value of 0.81means that 19% of the paths in the processmodelsmay actually represent behavior thatwas not recorded in the event log. It is an accept- able result compared to outcomes of artificial simulations from other scholars that use comparable metrics [38]. But it has to be validated in further research if this precision is high enough for the application in real world scenarios. The models produced by the MLPM from data set 1 to 3 were also analyzed using descriptive statistics. Figs. 5, 6, 7, and 8 show selected results for data sets 1 and 2. Figs. 5 and 7 show the frequency distributions of the mined process models depending on their model complexity measured as the number of included transitions. They show that the pro- cess models are distributed comparably to a normal distri- bution. Figs. 6 and 8 present scatter diagrams for data set 1 and 2. They illustrate the distribution of the number of rep- resented instance models in a process model depending on the model size. The diagrams show that the vast majority of process instances actually belong to very simple process models that contain only a few transitions. This observation confirms preliminary results from prior research work [56]. The diagrams for the third data set are not included due to space restrictions. They follow similar patterns. Selected descriptive statistical values are listed in Table 3 for all data sets.18 7 DISCUSSION The overall aim of the presented research is the improve- ment of financial audits by providing process mining solu- tions to discover process models from recorded event log data in ERP system. The main contribution of this paper is the introduction of a process mining algorithm that is able to discover the control flow and the data flow perspective in process models by simultaneously providing perfectly TABLE 2 Ppath 16 pPath Set 1 Set 2 Set 3 Mean 0.7925 0.8581 0.8034 Median 1 1 1 Minimum 0.1250 0.1165 0.2222 Maximum 1 1 1 Standard Deviation 0.2404 0.2229 0,2491 Variance 0,0578 0.0497 0.0620 Interquartile Range 0.4286 0.3000 0.4167 Fig. 5. Frequency distribution for data set 1. Fig. 6. Scatter diagram for data set 118. Fig. 7. Frequency distribution for data set 2. Fig. 8. Scatter diagram for data set 2. 16. Process models including loops were excluded from the calculation. 17. This is the case, for example, if too parallel branches like in Fig. 3a show different execution paths with PIG ¼ B ! C !ðf DÞ; ðC ! E ! DÞg but involving the same activities (in this case C). Then PPM ¼ B ! C ! Dð Þ; C ! E ! Dð Þ; B ! C ! E ! Dð Þ; ðC ! DÞf g withpPath ¼ 0:5. 18. The dependent variables in Figs. 6 and 8 use a logarithmic scaling. WERNER AND GEHRKE: MULTILEVEL PROCESS MINING FOR FINANCIAL AUDITS 829 fitting and highly precise models at different abstraction levels. The innovation of the presented artifact is achieved by combining already existing and newly developed meth- ods that lead to a novel solution for a new application area. The presented research followed a DSR approach. Gregor and Hevner [57] provide a useful framework for the catego- rization of knowledge generated by DSR and categorize research into four quadrants. The presented research work can be assigned to the exaptation quadrant because the main objective is to provide a solution for a new application area by partly using already existing knowledge. But it also affects the improvement quadrant by introducing a new method to model the control flow and data simultaneously in mined process models. Gregor and Hevner further differ- entiate between three levels of contribution types that range from abstract, complete and mature knowledge on the high- est level to more specific, limited and less mature knowl- edge on the lowest level. The research results presented in this paper are mainly located on the second level providing constructs and methods for the mining of process models and on the first level presenting an instantiated software artifact. The results that can be achieved by analyzing the mining outcomes can also be input for the third and highest knowledge contribution level. The distributions of the num- ber of instances over the number of transitions in Figs. 6 and 8 for example can lead to the assumption that their dis- tribution curves are very similar. But the descriptive statis- tics in Table 3 shows that the mean values for the number of transitions in the process model differ quite significantly from each other with 4.56 for data set 1, 4.95 for set 2 and 3.98 for set 3. It could hypothesized that the complexity of the mined process models relates to the maturity of the mined business processes following the assumption that a mature process is more integrated into information systems than a less mature. This research question surely needs further investigation, but it highlights how the presented results can be the starting point for further theoretical research. The presented mining algorithm is able to discover pro- cess models in accordance with the identified requirements to a large extent. The mined models are not absolutely pre- cise. It needs to be validated in further research if the achieved level of precision is sufficient in practice. Several other limitations need to be taken into account. The process models do not represent sound workflow nets according to commonly used definitions [3, p. 39]. This handicap is not too severe because the objective of process mining for finan- cial audits is the adequate modeling of the control and data flow perspective with precise and fitting process models. Formally well-structured process models are of minor inter- est. But soundness would be achieved if account places are neglected. The remaining models would then represent sound workflow nets but without representing the data flow perspective. The data sets were all extracted from SAP systems. It can therefore not be concluded that the research results also hold true for other data sources. But an impor- tant advantage of the used mining algorithm is its indepen- dence from the implemented data structures of a particular ERP system because it bases on the general structure of accounting entries. Some process models showed loops that occur when a transaction has cleared a journal item that was posted by the same transaction or by a transaction with the same transaction name located in the subsequent execu- tion path. This constellation leads to a deadlock in the pro- cess model which is not critical for the interpretation of the model but generally not desired for the modeling of correct process models. A solution could be the prevention of aggregating transitions carrying the same label if this would result in a loop. The mining algorithm produces precise and fitting pro- cess models at the cost of lacking generalization. It is there- fore not applicable for scenarios with highly variable business processes. In the worst case scenario all process instances show a different behavior. The mining algorithm would then not be able to aggregate any instance models and the set of process instances models would be identical to the set of process models resulting in no or little informa- tion gain. The data presented in Table 1 shows that this risk is not acute for the given application area. Business pro- cesses are usually standardized to a certain degree when they are operated via ERP systems. The data shows that the number of process models ranges from 307 for the smallest data set to 841 models for the largest. This may still seem to be a big number. But 63 process models in data set 1, for example, only consist of one transition. These represent triv- ial processes. The activities in these processes were mostly carried out by using a single general purpose transaction. They are of little interest from a process perspective but highly important from an audit perspective because this cat- egory of process models represent 96% of process instances in data set 1, 70% in set 2 and 51% in set 3. It is clear that fur- ther analytical procedures are necessary to address this cat- egory. A starting point could be the clustering of process models that use the same accounts. The process models that reflect the major business processes are those that contain many transitions and represent a high number of process instances. Data set 1 contains 135 process models consisting of 5 transitions. But just two of them already represent 62% of the instances of this category. It can therefore be assumed that the majority of instances for more complex process models only represent very infrequent behavior and can be tested traditionally by inspecting individual journal entries. The process models that represent many process instances and create a high value flow are interesting from a material- ity perspective and can be audited by including the testing of embedded application controls [58]. 8