2009 IEEE Conference on Commerce and Enterprise Computing

Process Mining of RFID-based Supply Chains

Kerstin Gerke, Alexander Claus
SAP Research, CEC Dresden, SAP AG
Dresden, Germany
kerstin.gerke@sap.com, alexander.claus@sap.com

Jan Mendling
Humboldt-Universität zu Berlin
Berlin, Germany
jan.mendling@wiwi.hu-berlin.de

***Abstract*—Process mining facilitates the analysis of business processes by extracting a process model from event logs. Most mining algorithms perform well on single-system event logs that explicitly refer to a process instance or case. However, in many operational environments such case identifiers are not directly recorded. In supply chain processes there are even further challenges, since different identification numbers and numerous aggregation steps prevent individual work items to become traceable as a case. In this paper, we investigate how the EPCglobal standard for processing Radio Frequency Identification (RFID) events can make supply chain data accessible for process mining. Our contribution is an algorithm that is able to deal with challenges of case identification and focus shifts. We present a prototypical implementation and use a process based on the Supply Chain Operations Reference (SCOR) Model to evaluate our implementation.**

***Keywords*-RFID; EPCglobal; process mining; supply chain**

I. Introduction

Delay of information processing between different parties and information aggregation pose a major challenge for supply chains. When real-time data on supply and demand is missing, the whole supply chain can be affected by the so-called bullwhip effect [12], i.e. small demand variation on the retailer side can trigger excessive demand variation for suppliers at earlier stages of the supply chain. The availability of accurate and timely information can help to anticipate this problem. In particular, information sharing between different supply chain participants is considered to be an effective strategy to circumvent the bullwhip effect [18].

Recently, there have been several IT innovations that drastically change the way how supply chains are managed. Most notably RFID technology and complementary concepts like the Internet of Things offer new mechanisms of accurate online information sharing. While most companies focus on streamlining operations with their RFID initiatives [11], RFID promises potential value for knowledge creation, decision support, and data mining. By providing precise data on product location and product characteristics, the collection of RFID data provides strategic value in discovery of new relationships and opportunities for process redesign [14].

However, although there is consensus about the need for global supply chain analysis, there has been little work on techniques for analyzing business processes that span a whole supply chain. A key issue in this context is that none of the supply chain parties has a comprehensive overview over the processes being executed. Process mining is a technique that automatically derives process models from a set of event logs. Its potential value for supply chain analysis relates to the discovery of interactions in a distributed processes [8], [22], which is a prerequisite for process improvement, and helpful for logistics network design and supply chain planning [14]. For example, it can reveal undesirable behavior that is not compliant with the specification of a predefined process. Process mining has been successfully applied to various intraorganizational problems [20], but the challenges of using supply chain event logs that span different companies is little understood so far.

This paper investigates how RFID technology and process mining can be combined for better supply chain analysis. We assume that the EPCglobal standard is used for exchanging RFID events. To be more concise, we provide a threefold contribution. First, we present an algorithm for deriving case identifiers from EPCglobal events. A preliminary version of this algorithm has been presented in [7]. In this way, we make process mining applicable. Second, we present a prototypical implementation as a validation of our concepts. This prototype takes RFID events and generates log files as a starting point for process mining. Third, we simulate a supply chain process using the Supply Chain Editor. Reconstructing the model with process mining based on the simulated logs allows us to assess the correctness of our approach.

The paper is structured accordingly. Section II gives an overview of RFID technology and the EPCglobal standard. We also illustrate the major challenges for making EPCglobal events applicable to process mining techniques and present a corresponding algorithm to derive case identifiers from EPCglobal events. Section III is dedicated to the validation of our approach. We introduce the Supply Chain Editor and a supply process inspired by the Supply Chain Operations Reference Model (SCOR). We then present the model resulting from process mining and discuss its relationship to the simulation model. Section IV relates our contribution to other research papers. Finally, Section V concludes the paper and gives an outline of future research.

978-0-7695-3755-9/09 $25.00 © 2009 IEEE
DOI 10.1109/CEC.2009.72

II. RFID EVENTS IN SUPPLY CHAINS

In this section, we investigate challenges of mining supply chain processes. In Section II-A, we give a brief overview of RFID and EPCglobal. Section II-B highlights the mining challenges. Section II-C describes the requirements for making EPCglobal events accessible for mining. Section II-D presents our approach to building cases from the events.

A. RFID technology and EPCglobal

RFID is an automatic identification technology that is used to track location and movements of physical objects. The usage of RFID requires different components such as tags, readers, middleware, and applications [9]. *Tags* are attached to physical objects and store at least a unique identifier. *Readers* are the hardware devices that read the information from tags. The task of the *middleware* is to provide the data in a suitable format to *RFID applications*. A major advantage of RFID in contrast to e.g. bar code is that the technology enables bulk scanning and rewriting [14].

Currently, international standards are emerging for the provisioning of RFID data across the internet. The most significant is the EPCglobal standard [5] describing the EPCglobal Network. The *EPCglobal Network* is an information network that will allow supply chain parties to easily access and exchange information about products. The access to product information is based on the *Electronic Product Code (EPC)* which uniquely identifies each product. The EPCs are stored by the *EPC Information Service (EPCIS)*.

EPCglobal standardizes four *event types*, shown in Fig. 1: object, aggregation, quantity, and transaction events. An *ObjectEvent* represents an event that relates to one or more EPCs. An *AggregationEvent* records the assignment of child EPCs to the parent EPC of the container. A *QuantityEvent* captures an event that relates to a specific quantity. A *TransactionEvent* represents an event in which EPCs become associated with business transactions, e.g. a delivery notification [5]. The event types contain *event fields* that can be divided into four dimensions of information: time, object, location, and business context. The *eventTime* stores the time an event is recorded at. The *epcList* is a list of EPCs naming the physical objects to which the event pertains. The identifier *bizTransList* denotes the specific business transactions, to which the EPCs are associated. The *parentID* represents an identifier of the parent of the EPCs given in the EPC list. The field *action* relates the event to the lifecycle of the related EPCs. The field can accept the values ADD, OBSERVE, or DELETE. When a pallet is created, the respective event is an AggregationEvent with action field ADD. The read point at which the event occurs is stored in the field *readPoint*. For further fields and detailed information on the EPCglobal standard, we refer to [5].

An event field refers either to a field defined in the EPCglobal specification or to a field defined as an extension. Extensions can be used for adapting the standard to particular situations in which new event fields are required for existing event types.

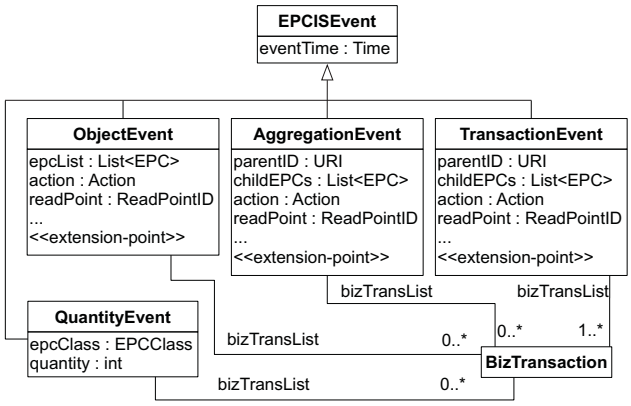

Figure 1. EPCISglobal event and fields

B. Challenges of mining supply chains

Process mining is a technique to analyze how processes are executed in reality. The input for process mining is a set of event logs, which all can be related to different instances of a process. These event logs typically stem from process-aware information systems such as workflow applications or ERP systems [10]. Using mining algorithms, it is possible to automatically construct process models from the event logs [8], [22]. In contrast to many manually created process models, the models generated by process mining give an accurate account of what is really happening in the company [20]. Such accurate models would be valuable information, in particular in a supply chain setting. However, there are two major reasons why the events that are generated by RFID applications cannot directly be used for process mining: case identification and focus shifts [7].

The problem with *case identification* results from the fact that EPCs are processed. This means that every event that can be logged only relates to EPCs - there is no explicit notion of a case identifier that groups events belonging to the same process instance. In order to make process mining applicable, we first have to assign every event to a process instance. Using an EPC as a process ID does not work in the general case because of the second problem.

The problem of *focus shifts* is illustrated in Fig. 2. It results from different containment relationships between EPCs (see [29]). Due to various packing and assembly operations, it is difficult to follow the object flow of a single process instance. At the shipper's site, there is first a packaging step that includes packing cartons on a pallet. An aggregation takes place by which the Serialized Global Trade Item Number (SGTIN) is associated with the Serial Shipping Container Code (SSCC). A SGTIN serves as an EPC associated with items. The SSCC is unique for a container. Aggregation events allows one to track which

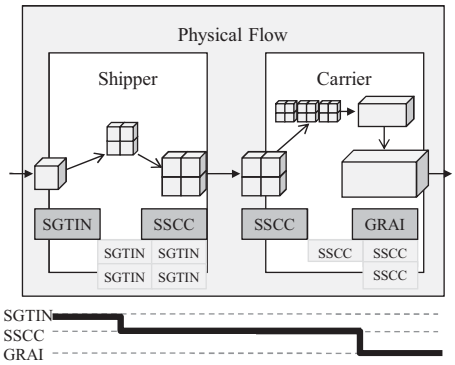

Figure 2. Packaging process

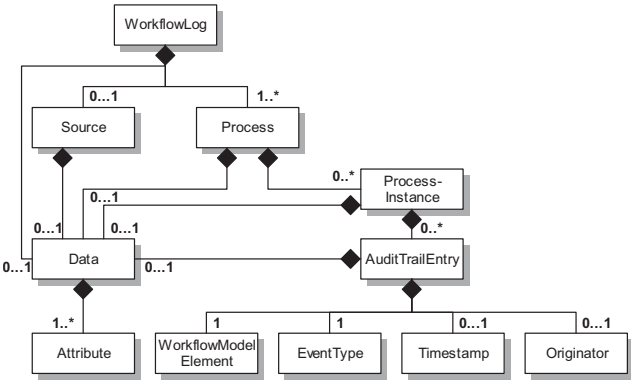

Figure 3. MXML format

exact products are packed into which container or pallet. After such an aggregation the focus shifts from the single product to the pallet. Depending on the material of the container, e.g. metal, it might not be possible to read the RFID tags of contained objects from outside. In the second packing step, the pallets are stuffed into a container identified by the Global Returnable Asset Identifier (GRAI). The GRAI contains all SSCCs providing the capability to identify and verify individual pallets. The monitoring of each containment stage of a product provides the supply chain parties with the ability to understand precisely which products, at a unique pack level, are shipped. This allows e.g. for accurate recall management.

C. The target format for process mining

In order to make events accessible to mining algorithms, we map the EPCglobal events into the Mining XML (MXML) format [24]. MXML is a generic XML-based format designed to store log files. It serves as input format for the process mining workbench ProM (www.processmining.org), which offers among others a variety of process mining algorithms.

The MXML format is depicted as an UML diagram in Fig 3. The root node of each log is the *WorkflowLog*, which can contain several *Processes* and optional information about the *Source* as well as further *Data*. Each *ProcessInstance* can have an arbitrary number of *AuditTrailEntries*. The AuditTrailEntries represent the events of the process. They are assumed to be in chronological order. The *WorkflowModelElement* describes the process activity. The *EventType* delineates the state change of the WorkflowModelElement. Every level of the hierarchy has the optional element *Data*, which can be used to store additional information. According to the MXML structure, we have to assign each event at least to a WorkflowModelElement and a ProcessInstance. Furthermore, we have to provide the Timestamp and the EventType [24]. There are several data fields of an EPCglobal event that can be directly mapped to the MXML fields. The mapping is illustrated in Tab. I.

D. Deriving cases from EPCglobal events

The main challenge is to assign events and the related EPCs to a process instance in such a way that the shifting focus between different assembled and disassembled business objects of varying granularity is handled appropriately.

For our approach, we use a concept that is similar to so-called *chained correlation* which is used in [21] to assign web service messages to process instances. The idea is to trace each object and use EPCs as case identifiers. This requires adding event entries for those items that are related to a particular EPC, but might not be readable because they are physically shielded in a container or have been merged with another object. There are two essential relationships that we have to track: aggregation and transformation. We consider two distinct types of aggregation: production and shipment aggregation [7]. *Production aggregation* is of a persistent nature: once different pieces are assembled they become a new object of its own. Production differs from *shipment aggregation*; for the latter the aggregation is of a temporary nature. In a way, it is transparent to the physical production process. Both types of aggregation can be distinguished based on EPCglobal events: shipment aggregations involve both an assembly event (*Aggregation ADD*) and a disassembly event (*Aggregation DELETE*) while production aggregations only imply an assembly event. The second type of relationship that has to be tracked is a *transformation*

Table I
MAPPING OF THE EVENT FIELDS

EPCISglobal *Event Type*	*Event Field*	**MXML** *Field*
All	eventClass & action	EventType
	eventTime	Timestamp
	readPoint	WorkflowModelElement
	readPoint	Originator
	bizTransList	Data
Object	epcList, sourceEPC	
Aggregation	parentID, childEPC	
Transaction	epcList, parentID	

relationship. In a transformation, one material is consumed for creating one (refinement) or more (split) respective outputs. This relationship is not directly visible according to the EPCglobal standard: an *Object DELETE* event occurs together with an *Object ADD* event. Yet, both events should be recorded at one particular read point, but we cannot make assumptions about the order of these both events. In the following, we assume that an EPCglobal extension field called *sourceEPC* is provided by the *Object ADD* event. In this way, we abstract from the question whether this field is inferred from the log or recorded by the EPCglobal implementation.

To find the EPCs affected by an event, we maintain a dependency graph holding at each point in time the current relations between EPCs while we replay the event log in chronological order. The definition of the graph is as follows. Each node represents an EPC. An aggregation relationship is represented by a directed arc from the parent EPC of the aggregate to all its child EPCs. A transformation relationship is represented by a directed arc from the EPC of the transformed product to its source EPC. To each EPC a set of events is attached. Thus, each node and its set of events is considered to represent a process instance. Since all relationships are of physical nature, they are hierarchically organized. An arc from an EPC epc_1 to an EPC epc_2 can be read as epc_2 *depends* on epc_1 in the sense that each event affecting epc_1 is also affecting epc_2. Due to the fact that in an event log many EPCs can occur autonomously, the resulting structure of the dependency graph is a forest, i.e. an unordered set of trees.

During the replay of the event log the dependency graph changes its structure. New nodes are established and arcs may be added or removed from the graph. For this reason the dependency graph cannot be constructed in advance but has to be interleaved with the assignment of events to EPCs. The events are propagated through all their descendants in the graph. This means that any event is not only attached to all EPCs occurring in the epcList or parentID field of the event but also to all EPCs which *depend* on them. The latter attachment ensures, that an assembly event affects both the aggregate and the contained product. Every event affecting a transformed product is also bound to its source product.

We define the algorithm in pseudo-code as in Fig. 4. Although we chose meaningful function names, we want to highlight the important aspects of the algorithm. In line 3, the transformation relation is processed. Note that we expect the event generator to be able to handle the EPCglobal extension field *sourceEPC*. Lines 14 to 16 account for the reversal of an existing aggregation relation. After assigning the event to all descendants the dependence towards parent and child EPC is dissolve by removing the arc between the depending EPCs from the graph.

Once the events are grouped into cases, the last step of the conversion is to write the MXML structure from the dependency graph. For each node in the graph a process instance in the MXML tree is created and all attached events are enclosed as child nodes to it. Grouping process instances into distinct processes identified by EPC classes allows us to mine for all products on their way through the supply chain as well as for specific product classes.

```
1:  for all Event e in EventChronology do
2:      if e.isObjectEvent then
3:          if e.isAddObject with sourceEPC then
4:              for all EPC epc in e.EPCs do
5:                  epc.addChild(sourceEPC)
6:          for all EPC epc in e.EPCs do
7:              epc.addEvent(e)
8:              epc.propagateToDescendants(e)
9:      else if e.isAggregationEvent then
10:         e.ParentEPC.addEvent(e)
11:         if e.isAddAggregation then
12:             e.ParentEPC.addChildren(e.ChildEPCs)
13:             e.ParentEPC.propagateToDescendants(e)
14:         else if e.isDeleteAggregation then
15:             e.ParentEPC.propagateToDescendants(e)
16:             e.ParentEPC.removeChildren(e.ChildEPCs)
17:     else if e.isTransactionEvent then
18:         e.ParentEPC.addEvent(e)
19:         e.ParentEPC.propagateToDescendants(e)
20:         for all EPC epc in e.EPCs do
21:             epc.addEvent(e)
22:             epc.propagateToDescendants(e)
```

Figure 4. Case construction

We have implemented this algorithm as a plugin for ProMImport framework as part of ProM offered by TU Eindhoven [23]. The plugin queries events from an EPCIS repository and generates MXML files. The query can be controlled by several parameters, e.g. time intervals, EPCs, locations etc. The resulting MXML file can be imported into ProM [24].

III. Validation

In this section, we validate our approach. Section III-A describes the simulation of EPCglobal events. Section III-B defines the types of transformations in the simulation model. Finally, Section III-C discusses the process model, which has been mined using the data generated by our implementation.

A. EPCglobal event generation

The tool we use for the generation of the EPCglobal events is called the *Supply Chain Editor*. The editor is based on a data model developed at the Cambridge University as part of the EU project BRIDGE (http://bridge-project.eu/). The events are stored in the open source EPCIS implementation Fosstrak (http://www.fosstrak.org/), which

implements the EPCglobal Network specification. The editor offers facilities to model and to simulate the item flow between locations (nodes) of a supply chain. There are different types of nodes that exhibit different behaviors (see Tab. II).

The behavior can be defined in more detail using fields like quantity, product, and delay which are not captured in this table. The *Producer* produces quantity items of a product. The *Observer* passes all items to the succeeding node in the chain. The *Assembler* assembles quantity items of different products to produce one item of a product. It is possible to differentiate between a temporary and a persistent assembly. The *Packer* produces a container item of type product with quantity items. The *Unpacker* unpacks the container item of a product. The *Fabricator* transforms a raw product into a refined product. All behaviors are allowed to associate or disassociate business transactions to the processed products.

Table II
EVENTS GENERATED BASED ON BEHAVIOR

Behavior	Icon	Event Type & Field ACTION	Event Field
Producer		Object ADD	$\emptyset$
Observer		Aggregation or Object OBSERVE	childEPCs
Assembler		Object ADD and DELETE, Aggregation ADD	childEPCs
Packer		Aggregation ADD	childEPCs
Unpacker		Aggregation DELETE	childEPCs
Fabricator		Object ADD and DELETE	sourceEPC

B. Supply chain model

For our validation, we use a supply chain model that is depicted in Fig. 5. The scenario represents the product flow along the supply chain according to the Supply Chain Reference Model (SCOR) [19]. The SCOR model is subdivided into five primary management processes on the top level (Plan, Source, Make, Deliver, and Return). We use the SCOR abbreviations as a reference, i.e. *S2*, *M2*, and *D2* for *Source*, *Make*, and *Deliver*, as well as *EM* indicating an *Enable* activity. In Fig. 5, the Make process is separated from the Delivery process by a dotted line. We assume that there are two second-tier suppliers (S2), a first-tier supplier (M2:1-M2:6, D2:1-D2:10), an original equipment manufacturer (OEM) (D2:13), and a shipper (D2:12) involved in the process.

For space limitations we only discuss the M2 process in detail. It includes focus shifts and case identification challenges that we need to cover in the validation. The category M2 describes a make-to-order production process. In our scenario, the M2 process starts with the coordination of all intermediate production activities to be performed in producing car doors (M2:1:ScheduleActivities). According to the production plan that results from M2:1 the sourced products are selected and physically moved from a stocking location to a specific point of production location (M2:2:IssueSourcedProduct). At the production location, a series of activities are performed to convert them from the semi-finished state to a state of completion (M2:3:Produce). Afterwards, the completed doors are sequenced onto returnable items (RTI) (M2:4:Package). The RTIs are allocated from a RTI pool (EM:RTIPool). They carry the doors until D.2.92 where they are replaced with OEM specific transportation means. The cleaned RTIs (EM:CleanRTI) are returned to the packaging site. The sequenced doors are moved into a temporary holding location to await movement to a finished goods location (M2:5:StageFinishedProduct). Finally, the release documentation is created (M2:6:ReleaseProducts).

Different kinds of focus shifts can be identified in this supply chain process.

1) *Shipment Aggregation:* In M2:4, a rack is loaded with doors and becomes the focus object.
2) *Shipment Disaggregation:* In D2:9, the focus shifts back to the doors at the moment the doors are unloaded from the rack.
3) *Production Aggregation:* In M2.312 a door handle is welded on a galvanized plate.
4) *Transformation:* Both transformation types are modeled in M2:33. A *split* takes place when a large piece of metal is cut into plates that are later assembled to the door. A *refinement* occurs when the door is galvanized which changes its value, but not its nature.
5) *Complex Processing:* A door consists of a door handle and a window. If a production error is detected, the faulty door handle has to be removed from the door for inspection. In M2:32 a spare part door handle is taken as replacement. Technically, this operation is a disaggregation of the faulty part and an aggregation including the replacement.

C. Process mining

We have implemented the case construction algorithm using the ProMImport facility of the ProM workbench. It has been extended for reading the input EPC events via the EPCIS capture interface and generating MXML, which is subsequently provided to the application ProM. In ProM, we use existing process mining algorithms to build the process model.

There are different aspects that we have to check in the validation: coherence, quantities, and transition reliability. Figure 6 shows the process model we have extracted with the Heuristics Miner plugin [25]. It can easily be confirmed that the resulting process model is *coherent*: the right branch matches the Deliver process and the left branch the Make

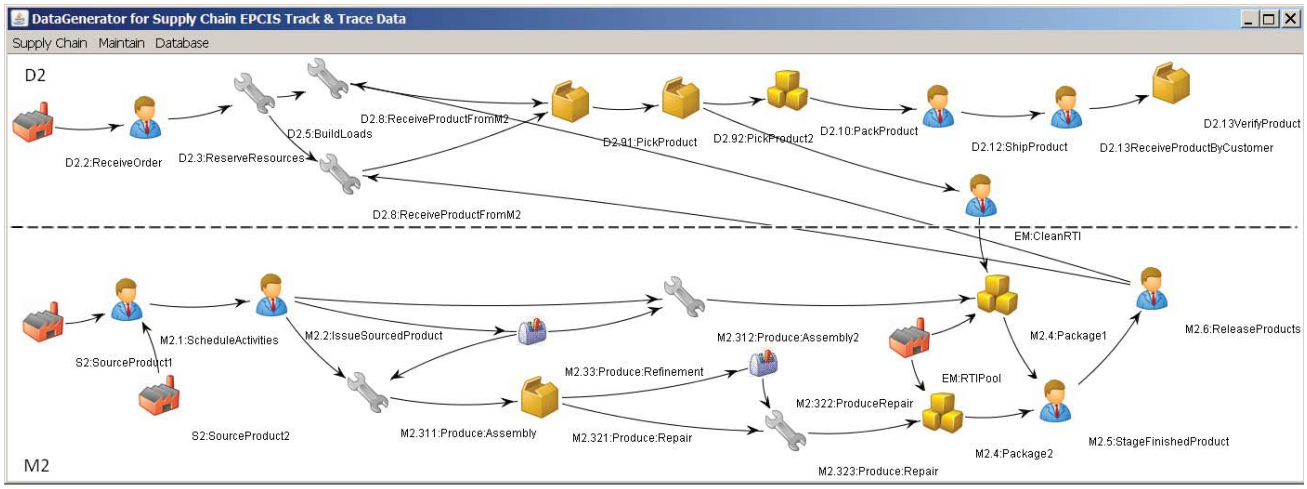

Figure 5. Supply chain simulator

process. In particular, all transitions of the simulation model have been reconstructed in the mined model. For instance, the direct transition from M2:2 to M2:312 represents the door handles that are assembled in M2:312 without an additional production activity. The plates need to be galvanized in the refinement process M2:33 before they can be assembled together with the handles into a car door.

The label of an arc indicates the frequency of an activity and the *quantities* of work items. For instance, in M2:311 a door handle and a galvanized plate are assembled into a car door. In our simulated sample of events, 51 galvanized plates from node M2.33 and 50 door handles from node M2.2 serve as an input for the assembly. The arc from M2.33 to M2.311 with frequency 49 represents the flow of the virtual source EPCs in order to reconstruct the transformation from a plate to a galvanized one. We can state that every assembly in M2:311 is composed of respectively 4 EPCs: one EPC (door), two child EPCs (handle, galvanized plate), and one source EPC (plate). In the graph, the frequency 200 in the node indicates that during the simulation 50 doors are assembled in M2.312. Since one door handle is missing for another production of a door, 1 of the 51 galvanized plates remains at the assembler's. Due to simulated delays in the production process, e.g. capacity constraints, only 49 doors (196 EPCs) are passed on to the node M2:311. One door remains in node M2.311. In this way, the quantities in the model directly relate to the settings of the simulator.

The squared boxes represent activities and the arcs between these activities represent dependency relations measured as *reliability*. This dependency measure on the arcs indicates the reliability of each causal relation. The closer the value comes to one the more reliable the dependency relation between the connected activities is likely to be. For example, every door handle and plate being produced in S2 arrive at M2:1. The reliability of respectively 0.998 expresses that we can be confident that the dependencies between the two activities exist. Since some items cannot be packed due to minimum batch sizes greater one, the reliability does not equal 1 but values close to it. We conclude that the process model shown in Fig. 6 confirms that our algorithm handles EPCglobal events properly, in particular bundle, transformation, and disassembly operations.

IV. Related work

Our work can be related to different streams of research in the automatic identification technology, web services and business process management domain.

Several works have demonstrated that classical process mining can be applied to web service interaction mining if unique case identifiers are included in each log entry (see [4], [13]). Once such information is not available, cases have to be identified based on other clues. This challenge is not only apparent for mining, but also at run-time. Several executable process modeling languages address this problem by offering so-called correlation sets. A correlation set is essentially a query that retrieves identifiers from messages that are unique for a particular process instance. The correlation set concept is included in BPEL [1] and BPMN [15] for dispatching messages to the correct process instance at runtime. The correlation problem has been discussed theoretically in [2], [3]. For mining web service interactions, the authors of [21] define a concept of chained correlation: a message can be assigned to a case via its preceding message, the latter via its own preceding message, and so forth. In [16], the authors use an algorithm to identify correlation relevant fields that can be used to construct case identifiers. In [17], the authors define an algorithm to construct a finite state machine of business protocols from logs without case identifiers.

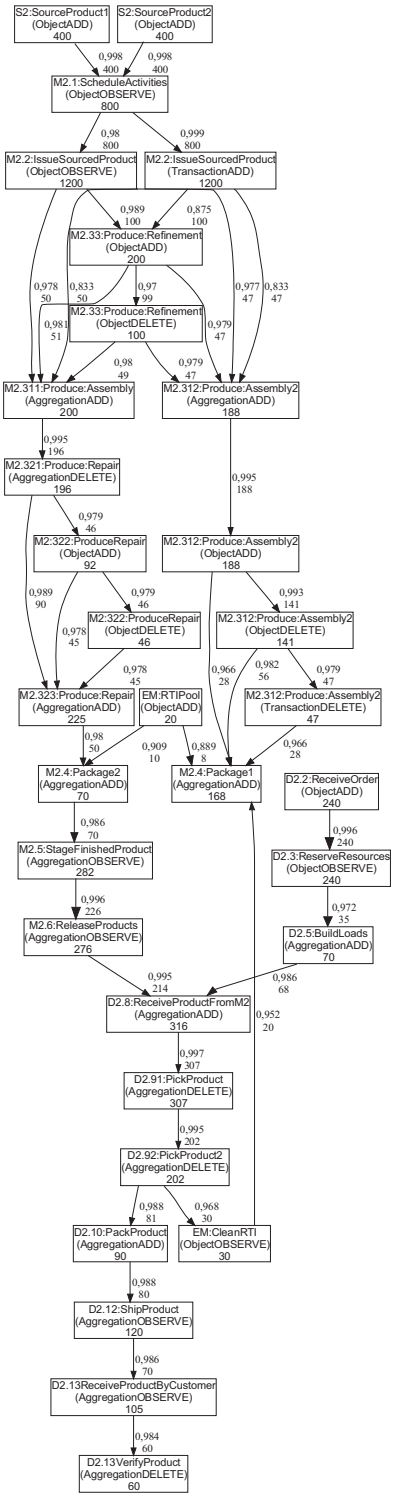

Figure 6. Process model derived by process mining techniques

Our algorithm also builds upon the observation that certain identifiers, namely EPCs, belong to a unique process instance. This has previously been used in [27] to mine workflows based on RFID data (but abstracting from focus shifts and containment relationships). The general idea of replicating events of objects for its contained objects is somewhat related to chained correlation of [21] and querying containment relationships [28]. More specifically, we reconstruct the correlation relationships that are implicitly given in the EPCglobal events. Furthermore, and as a distinction from the previously mentioned works, we consider physical flows with logistic bundling operations that cannot be uniquely assigned to one particular physical item.

Our contribution can also be related to practical challenges of process mining. There is a growing body of knowledge that reports on case studies in different application domains. In [26], the authors have discussed the potentials of process mining in supply chains. However, the discovery of distributed processes is based on the assumption that there is a common point of reference for all involved parties, e.g. an order number. While there are considerable achievements [20], several problems still need to be solved [6]. A particular problem relates to the reformatting and enrichment of log data that stems from information systems that are not directly process-aware, e.g. SAP log data [10]. Our work is unique in this context, as it identifies how RFID events (available in EPCglobal format) can be used to construct cases such that process mining can be applied.

V. Conclusion

In this paper, we identified case identification and focus shifts as the major challenges for making process mining applicable on supply chain events. We showed how the EPCglobal standard for processing RFID can be utilized to construct case information from such events. Our contribution is a corresponding algorithm along with a prototypical implementation as part of the ProM workbench. We further evaluated our approach using a car manufacturer scenario which is based on the Supply Chain Operations Reference Model (SCOR). In this way, we were able to demonstrate that the mined process models are coherent, that they show the expected transition reliability as well as the expected quantities.

At this stage our research has some limitations. So far, we did not have access to real-world EPCglobal events. We are currently negotiating with different companies about applying our mining tools on real data. In particular, we aim to learn from such a case study how well our approach scales with huge amounts of data and in how far the mining results are informative to supply chain managers. We assume that browsing cases at different levels of granularity is a helpful technique for supporting decision making. Currently, we are working on respective concepts and tools.

REFERENCES

[1] A. Alves de Medeiros et al., "Web services business process execution language version 2.0", 2007, OASIS, 2007.

[2] A. Barros, G. Decker, M. Dumas, and F. Weber, "Correlation patterns in service-oriented architectures", In M. Dwyer and A. Lopes, editors, *Fundamental Approaches to Software Engineering*, vol. 4422, pp. 245–259. Springer, 2007.

[3] G. Decker and J. Mendling, "Process instantiation", *Data and Knowledge Engineering*, 2009.

[4] S. Dustdar and R. Gombotz, "Discovering web service workflows using web services interaction mining", *Int. J. of Process Integr. and Management*, 1(4):256–266, 2006.

[5] "EPCglobal Inc., EPC Information Services (EPCIS) Version 1.0.1 Specification", http://www.epcglobalinc.org, 2007.

[6] M. Genrich, A. Kokkonen, J. Moormann, M. zur Muehlen, R. Tregear, J. Mendling, and B. Weber, "Challenges for business process intelligence: Discussions at the BPI Workshop", In A. ter Hofstede, B. Benatallah, and H.-Y. Paik, editors, *Proc. of the BPM 2007 Workshops*, vol. 4928 of *Lecture Notes in Computer Science*, 2008. Springer.

[7] K. Gerke, J. Mendling, and K. Tarmyshov, "Case construction for mining supply chain processes", In W. Abramowicz, editor, *Proc. of the Conf. on Business Information Systems*, Springer, 2009.

[8] G. Greco, A. Guzzo, G. Manco, and D. Saccà, "Mining and reasoning on workflows", *IEEE Transactions on Knowledge and Data Engineering*, 17(4):519–534, 2005.

[9] O. Günther, W. Kletti, and U. Kubach, *"RFID in Manufacturing*", Springer, 2008.

[10] J. Ingvaldsen and J. Gulla, "Preprocessing support for large scale process mining of SAP transactions", In A. ter Hofstede, B. Benatallah, and H.-Y. Paik, editors, *Proc. of the BPM 2007 Workshops*, vol. 4928, 2008. Springer.

[11] L. Ivantysynova, *"RFID in manufacturing: Mapping the shop floor to IT-enabled business processes"*, PhD thesis, Humboldt-Universität zu Berlin, 2008.

[12] H. Lee, V. Padmanabhan, and S. Whang, "The bullwhip effect in supply chains", *Sloan Management Rev.*, pp. 93–102, 1997.

[13] H. R. M. Nezhad, R. Saint-Paul, B. Benatallah, and F. Casati, "Protocol discovery from imperfect service interaction logs", In *ICDE*, pp. 1405–1409. IEEE, 2007.

[14] F. Niederman, R. G. Mathieu, R. Morley, and I.-W. Kwon, "Examining RFID applications in supply chain management", *Communications of the ACM*, 50(7):92–101, 2007.

[15] Object Management Group, "Business Process Modeling Notation (BPMN) Specification", OMG, February 2006.

[16] W. D. Pauw, R. Hoch, and Y. Huang, "Discovering conversations in web services using semantic correlation analysis", In *Intl. Conf. on Web Services (ICWS 2007)*, pp. 639–646. IEEE Computer Society, 2007.

[17] B. Serrour, D. P. Gasparotto, H. Kheddouci, and B. Benatallah, "Message correlation and business protocol discovery in service interaction logs", In Z. Bellahsene and M. Léonard, editors, *Adv. Information Systems Engineering, 20th International Conference*, vol. 5074, pp. 405–419. Springer, 2008.

[18] A. Sharma, A. Citurs, and B. Konsynski, "Strategic and institutional perspectives in the adoption and early integration of Radio Frequency Identification (RFID)", In *Hawaii International Conference on Systems Sciences*, vol. 40, page 3755. IEEE, 2007.

[19] Supply-Chain Council, "Supply Chain Operations Reference Model, SCOR", Version 8.0, 2006.

[20] W. van der Aalst, H. Reijers, A. Weijters, B. F. van Dongen, A. Alves de Medeiros, M. Song, and H. Verbeek, "Business process mining: An industrial application", *Information Systems*, 32(5):713–732, 2007.

[21] W. van der Aalst, M. Dumas, C. Ouyang, A. Rozinat, and E. Verbeek, "Conformance checking of service behavior", *ACM Trans. Internet Techn.*, 8(3), 2008.

[22] W. van der Aalst, T. Weijters, and L. Măruşter, "Workflow mining: Discovering process models from event logs", *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.

[23] B. F. van Dongen, A. Alves de Medeiros, H. Verbeek, A. Weijters, and W. van der Aalst, "The ProM framework: A new era in process mining tool support", In G. Ciardo and P. Darondeau, editors, *Applications and Theory of Petri Nets 2005*, vol. 3536, pp. 444–454, 2005. Springer.

[24] B. F. van Dongen and W. van der Aalst, "A meta model for process mining data", In *Proc. of the CAiSE'05 Workshops*, vol. 2, pp. 309–320, 2005.

[25] A. Weijters, W. van der Aalst, and A. K. Alves de Medeiros, "Process Mining with the Heuristics Miner Algorithm", *Beta Working Paper Series*, WP 166, Eindhoven University of Technology, Eindhoven, 2006.

[26] L. Măruşter, J. C. Wortmann, A. J. M. M. Weijters and W. van der Aalst, "Discovering distributed processes in supply chains", *Proc. of the International Conference on Advanced Production Management Systems*, pp. 119–128, 2002.

[27] H. Gonzalez, J. Han, J. and X. Li, "Mining compressed commodity workflows from massive RFID data sets", *CIKM '06: Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, ACM, pp. 162–171, 2006.

[28] D. Lin, H. Elmongui, E. Bertino and B. Ooi, "Data management in RFID applications", *Proc. of the International Conference on Database and Expert Systems Applications*, pp. 434–444, 2007.

[29] F. Wang and P. Liu, "Temporal management of RFID Data", *Proc. of the 31st International Conference on Very large Data Bases, VLDB Endowment*, pp. 1128–1139, 2005.