

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

SAP AMERICA, INC.,
Petitioner,

v.

CELONIS SE,
Patent Owner.

IPR2027-00004
Patent: US 11,256,708
Issued: February 22, 2022
Application No. 16/256,380
Filed: January 24, 2019

Title: METHOD OF CREATING PROCESS PROTOCOLS

PETITION FOR
INTER PARTES REVIEW OF U.S. PATENT NO. 11,256,708

TABLE OF CONTENTS

	Page
LIST OF EXHIBITS	v
MANDATORY NOTICES UNDER 37 C.F.R. § 42.8	vii
CLAIMS LISTING	ix
I. INTRODUCTION	1
II. GROUNDS FOR STANDING PER 37 C.F.R. § 42.104(a)	2
III. IDENTIFICATION OF CHALLENGE	2
IV. LEVEL OF ORDINARY SKILL IN THE ART	2
A. Person Of Ordinary Skill In The Art (“POSITA”)	3
B. State Of The Art	3
1. Schemas And Keys Were Regularly Used To Define Relationships Between Database Tables	3
2. It Was Common For Different Systems To Execute Multiple Related Processes That Generate Process Data	4
V. THE ’708 PATENT	4
A. The Prosecution History	5
VI. CLAIM CONSTRUCTION	7
VII. GROUND 1: CLAIMS 1-14 ARE OBVIOUS OVER RINKE937, NOOIJEN, AND HARRISON	8
A. Rinke937 (EX1004)	8
B. Nooijen (EX1005)	10
C. Harrison (EX1006)	13
D. Combining Rinke937, Nooijen, And Harrison	16
E. Claim 1	22

1.	Element [1.1].....	22
	a) Element [1.1.a].....	22
	b) Element [1.1.b]	29
	c) Element [1.1.c].....	30
	d) Element [1.1.d]	33
2.	Element [1.2].....	33
3.	Element [1.3].....	35
4.	Element [1.4].....	39
5.	Element [1.5].....	41
6.	Element [1.6].....	44
7.	Element [1.7].....	46
8.	Element [1.8].....	49
F.	Claim 2	52
G.	Claim 3	53
H.	Claim 4	54
I.	Claim 5	56
J.	Claim 6	57
K.	Claim 7	60
L.	Claim 8	62
M.	Claim 9	63
N.	Claim 10	65
O.	Claim 11	66
P.	Claim 12	67
Q.	Claim 13	68

R. Claim 14 69

VIII. NO OBJECTIVE INDICIA OF NON-OBVIOUSNESS.....70

IX. CONCLUSION.....70

CERTIFICATE OF COMPLIANCE.....71

CERTIFICATE OF SERVICE 72

TABLE OF AUTHORITIES

Page(s)

Cases

Phillips v. AWH Corp.,
415 F.3d 1303 (Fed. Cir. 2005) 7

Board Decisions

Amazon.com, Inc. v. Vocalife LLC,
IPR2021-01331, Paper 11 (Jan. 31, 2022) 11

Statutes

35 U.S.C. § 103 2

Other Authorities

83 Fed. Reg. 51340 7

Rules

37 C.F.R. § 42.104 2
37 C.F.R. § 42.24 71
37 C.F.R. § 42.6 72

LIST OF EXHIBITS

No.	Description
1001	U.S. Patent No. 11,256,708 (“ 708 patent ”)
1002	File History of U.S. Patent No. 11,256,708
1003	Declaration of Dr. Craig Knoblock, dated October 21, 2026 (“ Knoblock Decl. ” or “ Knoblock ”)
1004	U.S. Patent App. Pub. No. 2017/0212937 to Rinke et al. (“ Rinke937 ”)
1005	Erik H. J. Nooijen, Boudewijn F. van Dongen and Dirk Fahland, <i>Automatic Discovery of Data-Centric and Artifact-Centric Processes</i> , BPM 2012 Int’l Workshops, p. 316 (Marcello La Rosa & Pnina Soffer eds., 2013) (“ Nooijen ”)
1006	U.S. Patent App. Pub. No. 2018/0096000 to Harrison et al. (“ Harrison ”)
1007	Martti Kantola et al., <i>Discovering Functional and Inclusion Dependencies in Relational Databases</i> , Int’l J. of Intelligent Sys., vol. 7, p. 591 (1992) (“ Kantola ”)
1008	Yue Zhuge et al., <i>View Maintenance in a Warehousing Environment</i> , Procs. of the ACM SIGMOD Int’l Conf. on Mgmt. of Data, Ass’n for Computing Mach., p. 316 (1995) (“ Zhuge ”)
1009	Meihui Zhang et al., <i>On Multi-Column Foreign Key Discovery</i> , Procs. of the VLDB Endowment, vol. 3, no. 1, p. 805 (2010) (“ Zhang ”)
1010	Wil van der Aalst et al., <i>Process Mining Manifesto</i> , Bus. Process Mgmt. Workshops, BPM 2011, Lecture Notes in Bus. Info. Processing, vol. 99, p. 169, Springer-Verlag (2012) (“ Process Mining Manifesto ”)
1011	Claims Listing for U.S. Patent No. 11,256,708
1012	Ramez Elmasri and Shamkant B. Navathe, <i>Fundamentals of Database Systems</i> , 7th ed., Pearson (2016) (“ Elmasri ”) (Excerpt)

No.	Description
1013	Oracle® Database, <i>Database Concepts</i> (Vol. 18c) (Aug. 2018) (“ Oracle ”) (Excerpt)
1014	D.A.M. Piessens, <i>Event Log Extraction from SAP ECC 6.0</i> , Master’s thesis, Eindhoven Univ. of Tech., Dep’t of Mathematics and Comput. Sci., Apr. 30, 2011 (“ Piessens ”)
1015	Xixi Lu et al., <i>Discovering Interacting Artifacts from ERP Systems</i> , IEEE Transactions on Servs. Computing, vol. 8, no. 6, p. 861 (Nov./Dec. 2015) (“ Lu ”)
1016	U.S. Patent App. Pub. No. 2017/0068705 (“ Rinke705 ”)
1017	Jan Claes and Geert Poels, <i>Merging Event Logs for Process Mining: A Rule Based Merging Method and Rule Suggestion Algorithm</i> , Expert Sys. with Applications, vol. 41, no. 16, p. 7291 (2014) (“ Claes ”)
1018	Wil M.P. van der Aalst, “ <i>Intra- and Inter-Organizational Process Mining: Discovering Processes Within and Between Organizations</i> ,” The Practice of Enterprise Modeling, 4th IFIP WG 8.1 Working Conf., PoEM 2011, Oslo, Norway, Nov. 2–3, 2011 (“ van der Aalst ”)
1019	U.S. Patent App. Pub. No. 2017/0213167 to Rinke et al. (“ Rinke167 ”)

MANDATORY NOTICES UNDER 37 C.F.R. § 42.8

1. Real Party In Interest

The real parties in interest (“RPIs”) are SAP America, Inc. and SAP SE.

In an abundance of caution, Petitioner also identifies each of the following entities as an RPI for purposes of this proceeding only, without conceding that any satisfies the flexible, multi-factor, highly fact-dependent legal test for RPI: SAP Deutschland SE & Co. KG; SAP Beteiligungs GmbH; and SAP Labs, LLC. Each of these three entities is related to RPI SAP SE.

2. Related Matters

The ’708 patent (EX1001) has been asserted in the following district court litigation (the “district court case”): *Celonis SE, et al. v. SAP SE and SAP America, Inc.*, Case No. 2:25-cv-01037 (E.D. Tex.), filed October 10, 2025.

3. Lead And Back-Up Counsel, And Service Information

Lead Counsel	Back-up Counsel
Andrew M. Mason, Reg. No. 64,034 andrew.mason@klarquist.com	Amy Haspel, Reg. No. 78,385 amy.haspel@klarquist.com Samuel B. Thacker, Reg. No. 78,633 samuel.thacker@klarquist.com Frank Morton-Park, Reg. No. 80,750 frank.morton-park@klarquist.com
KLARQUIST SPARKMAN, LLP 121 SW Salmon Street, Suite 1600 Portland, Oregon, 97204 503-595-5300 (phone) 503-595-5301 (fax)	

Petitioner consents to service via email at the above email addresses and the email address of SAP-CelonisIPR@klarquist.com.

Pursuant to 37 C.F.R. § 42.10(b), concurrently filed with this Petition is a Power of Attorney executed by Petitioner and appointing the above counsel.

Petitioner authorizes Account No. 02-4550 to be charged for any fees, including those enumerated in 37 C.F.R. § 42.15.

CLAIMS LISTING

Representative claim 1 is listed here, with numbering added in the left-hand column to indicate claim elements:

[1.1.a]	A computer-implemented method of creating a process protocol in a local computer system (LC), which comprises a processor and a storage device, wherein the process protocol encodes object related activities and is created from raw data, which is stored in part in a first external computer system (EC1) and in part in a second external computer system (EC2),
[1.1.b]	wherein the raw data in the external computer systems is stored in a number of data tables,
[1.1.c]	wherein the raw data comprises data, which is created during the execution of processes in the first external computer system and in the second external computer system,
[1.1.d]	wherein a process comprises several process steps, wherein each process step creates the data during execution, and wherein the method comprises:
[1.2]	(a) importing the data tables from the first external computer system and from the second external computer system into the local computer system, and storing the imported data tables in the storing device of the local computer system,
[1.3]	(b) in the local computer system, determining and reconstructing the relationships between the imported data tables,
[1.4]	(c) in the local computer system, determining and selecting a first data table from the number of imported data tables, wherein each data record stored in the first data table represents an object for which the process protocol is to be created,
[1.5]	(d) in the local computer system, determining and selecting a number of second data tables from the number of imported data tables, which are necessary for the creation of the process protocol, based on the selected first data table and based on the reconstructed relationships between the imported data tables, wherein each data record stored in the number of second data tables represents a process step,
[1.6]	(e) in the local computer system, determining activities for the objects of the selected first data table based on the selected second data tables and the reconstructed relationships between the selected first data table and the selected second data tables,

[1.7]	(f) in the local computer system, creating an executable transformation, which based on [i] the selected first data table, [ii] the selected second data tables, and [iii] the relationships between the selected first data table and the selected second data tables [iv] is adapted to extract data from the first data table and from the second data tables based on the determined activities, and to store the extracted data as process protocol, and
[1.8]	(g) in the local computer system, executing the transformation, wherein during execution of the transformation the data is extracted from the data tables stored in the external computer systems (ES1, ES2), and storing the extracted data as process protocol, and providing the created process protocol to a Process Mining System for analysis and reconstruction of the processes carried out in the first external computer system and in the second external computer system.

A list of all challenged claims is provided in EX1011 and, for reference, the claims are also listed throughout the petition.

I. INTRODUCTION

SAP America, Inc. (“SAP” or “Petitioner”) respectfully requests *inter partes* review (“IPR”) of claims 1-14 (collectively, the “challenged claims”) of U.S. Patent No. 11,256,708 (EX1001, “the ’708 patent”), allegedly owned by Celonis SE (“Patent Owner” or “PO”).

The challenged claims relate to creating an event log (recited as a “process protocol”) from raw data stored on multiple computers. Sole independent claim 1 involves importing data tables, determining and reconstructing relationships between those imported tables, and using the relationships to create an “executable transformation.” The transformation is then executed to extract event data from tables and create an event log (“process protocol”) for process-mining analysis. During prosecution, Applicant argued that the claims required determining unknown relationships after importing the tables and using these relationships in creating the claimed executable transformation, and that the prior art lacked these features. EX1002, 192-198/228.

Yet Nooijen (EX1005), not of record during prosecution, teaches automatically determining unknown table relationships in a process for event log generation. And Harrison (EX1006), also not of record, teaches importing data tables from multiple source systems to discover previously unknown relationships between those tables. Rinke937 (EX1004), in turn, discloses a basic framework for creating

and using executable transformations that generate event logs from raw process data stored in source systems. Combined, the complementary teachings of Rinke937, Nooijen, and Harrison render each challenged claim unpatentable.

II. GROUND(S) FOR STANDING PER 37 C.F.R. § 42.104(a)

Petitioner certifies that the '708 patent is available for IPR and that Petitioner is not barred or estopped from requesting an IPR challenging the patent claims on the grounds identified in this petition.

III. IDENTIFICATION OF CHALLENGE

Petitioner requests IPR of claims 1-14 of the '708 patent, on the following statutory ground:

	Reference(s)	Basis	Claims
Ground 1	Rinke937 (EX1004), Nooijen (EX1005), and Harrison (EX1006)	§ 103	1-14

In Section VII below, the Petition presents evidence of unpatentability and establishes a reasonable likelihood that Petitioner will prevail in establishing that each challenged claim is unpatentable.

IV. LEVEL OF ORDINARY SKILL IN THE ART

This section defines the level of ordinary skill in the art and explains certain aspects of the state of the art as of January 24, 2019, the effective filing date of the challenged claims.

A. Person Of Ordinary Skill In The Art (“POSITA”)

A POSITA at the relevant time would have had a bachelor’s degree in computer science, information systems, data science, or similar discipline, and approximately two years of academic, research, or industry experience with process (workflow)-mining, data-mining, or similar systems for analyzing event, log, or process data. Additional education could substitute for professional experience and vice versa. EX1003, ¶¶25-27.

B. State Of The Art

**1. Schemas And Keys Were Regularly Used
To Define Relationships Between Database Tables**

Relational database schemas were conventionally used to define both the structure of database tables and the relationships among those tables. A 2016 undergraduate textbook explained that a table’s “relation schema” specifies the table name and “a list of attributes.” EX1012, 151-152. It further explained how foreign-key constraints link records in different tables through matching key values to reflect “relationships among the entities represented by the relation schemas.” EX1012, 163-164. Likewise, keys “describe the relationships between the tables and columns of a relational database.” EX1013, 5-3; *see also* EX1014, 10; EX1003, ¶28.

2. It Was Common For Different Systems To Execute Multiple Related Processes That Generate Process Data

The Process Mining Manifesto (hereinafter, “the Manifesto”) recognized in 2012 that event data for a process may be “distributed over a variety of sources.” EX1010, 14-16; *see also* EX1018, 1-2 (2011 process mining paper noting that “many processes are distributed over multiple organizations and different organizations are executing similar processes”); EX1016, [0015]-[0017], [0103], Fig. 1; EX1017, 2; EX1001, 1:43-49 (describing, in background, that “in practice,” different “partial processes” of an overall process are executed on different computer systems). POSITAs thus knew that analysis techniques needed to handle these types of overarching (e.g., “cross-organizational”) business processes. EX1003, ¶¶29-30.

V. THE ’708 PATENT

The ’708 patent issued February 22, 2022, from an application filed January 24, 2019. The alleged invention “relates to a method of creating process protocols” from data stored “on different computer systems.” EX1001, 1:6-9. The patent uses the term “process protocol” to refer to the “processed form” of information extracted from data created by computer processes such as “business processes” or “technical processes.” *Id.*, 1:32-37; 1:13-20.

The ’708 patent admits it was “known” to use such processed information “to analyze processes” (*id.*, 1:21-30) but alleges that such analysis could only be run on

the computer where the data was generated (*id.*, 1:38-42) and was unable to analyze processes that were executed on multiple computer systems (*id.*, 1:61-66).

As an example, it describes a technical process where “information relating to the assembly process may be stored in a first computer system, and information relating to a subsequent technical test process may be stored in a second computer system.” *Id.*, 1:57-60. The ’708 patent admits these sub-processes could separately “be analyzed by means of methods known from prior art,” but alleges that an analysis of the overall process “is not possible.” *Id.*, 1:61-66.

The ’708 patent accordingly proposes creating a single process protocol from raw process data distributed across multiple external computer systems. In one example, the process protocol is created from data stored in two computer systems, one system that stores “order information” and “order history” and another that stores related “call” data. *Id.*, 11:29-12:9.

A. The Prosecution History

The Examiner first rejected the pending claims over various combinations of US2017/0068705 (“Rinke705”), US2017/0213167 (“Rinke167”), and US2011/0178974. EX1002, 105-117/228. Applicant responded by amending claim 1 to add “determining and” before “reconstructing the relationships between the imported data tables.” *Id.*, 143/228.

In remarks, Applicant explained that “a process protocol” refers to “data in processed form” and admitted that “the processing of data into process protocols is known.” *Id.*, 148/228. It further explained that analyzed processes might run on “multiple external computer systems,” such that a process protocol might “comprise separate data created on multiple external computer systems.” *Id.*, 149/228. Applicant alleged that the prior art could independently analyze “the partial processes” executed on each computer system, but not the “overall process.” *Id.*

Applicant explained that, prior to the “determining and reconstructing” step, relationships between imported tables “may not be known, as they are sourced from different computer systems,” and expressly characterized the amendment as “clarify[ing] the unknown nature of such relationships.” *Id.*, 149/228.

Applicant acknowledged similarities between the event logs in Rinke167 and the claimed process protocol, stating that their data structures “may show parallels.” *Id.*, 152/228. But it argued that the Rinke167 event log was “not created by a method similar to that claimed,” purportedly because in Rinke167 “the relationships between different data tables” were predefined and “not determined or reconstructed in the local computer system after acquiring the imported data tables from the external computer system.” *Id.*

After a final Office action maintaining substantially the same prior art rejections, Applicant filed an RCE and further amended claim 1 to incorporate an

external-table extraction step previously recited in claim 3. *Id.*, 157-172, 186, 190/228. Applicant again distinguished Rinke167 because its table relationships were predefined within process sensors: “all such relationships are already known and would not need to be reconstructed.” *Id.*, 196-197/228. Applicant also distinguished using “predetermined process sensors” from creating an executable transformation based on imported tables and determined relationships. *Id.*, 192, 198/228. Although Applicant acknowledged that Rinke167 “relates to the transformation of raw data from the IT source system using predetermined process sensors” and that “new data would be transformed using the same process sensors,” Applicant maintained that “[t]here is no teaching of the creation of an executable transformation followed by the application of that transformation to data tables stored in the external computer systems.” *Id.*, 198/228.

The Examiner then allowed the claims without any meaningful discussion of the prior art. *Id.*, 206/228.

VI. CLAIM CONSTRUCTION

Unless otherwise noted below, Petitioner applies the plain and ordinary meaning of all claim terms. 83 Fed. Reg. 51340; *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312-1313 (Fed. Cir. 2005) (*en banc*).

The term “**determining and reconstructing the relationships between the imported data tables**” should be construed as “after importing the data tables,

determining and reconstructing relationships between the imported data tables, where the relationships between tables imported from different computer systems were unknown prior to the determining and reconstructing.” EX1002, 149, 152, 196-97/228.

**VII. GROUND 1: CLAIMS 1-14 ARE OBVIOUS
OVER RINKE937, NOOIJEN, AND HARRISON**

As explained below, claims 1-14 are obvious over the combined teachings of Rinke937, Nooijen, and Harrison. EX1003, ¶40.

A. Rinke937 (EX1004)

U.S. Patent Application Publication No. 2017/0212937 to Rinke et al. (“Rinke937”) published July 27, 2017, and is prior art under 35 U.S.C. § 102(a)(1). EX1004, cover; EX1003, ¶41. Although it shares some disclosure with Rinke167 (considered during prosecution), Rinke937 is directed to a different alleged invention and contains different claims and additional disclosure directed to event log generation (*compare* EX1004, [0012]-[0064] *with* EX1019, [0012]-[0065]).

Rinke937 describes methods for “sens[ing] process steps and build[ing] up event logs based on raw data stored in a source system.” EX1004, [0010]. These methods “transform the source data into an event log format for further process mining analysis.” *Id.*; *id.*, [0002]-[0009]; [0072] (“[R]aw data from the IT source system is transformed into a representation which is capable for process mining

applications.”). These event logs allow process mining tools to reconstruct and analyze the actual process executed in the source system. *Id.*, [0088]; EX1003, ¶42.

Rinke937’s event logs include process element identifiers and process steps associated with each element, and reflect the order of steps, e.g., by timestamp. *Id.*, [0012]-[0023], [0075]-[0086]. Rinke937 provides example event logs, such as the following:

Unique ID	Process Step	Timestamp (Order)
1	Purchase Requisition	2012-01-01 13:05
1	Purchase Requisition Approval	2012-01-01 15:09
1	Purchase Order Item	2012-01-02 17:03
1	Goods Receipt	2012-01-03 13:01
1	Invoice Receipt	2012-01-03 15:00
2	Purchase Requisition	2012-01-05 11:00
2	Purchase Requisition Approval	2012-01-05 17:00
...		

Id., [0086]; EX1003, ¶43.

In Rinke937, the transformation from raw data to event logs involves units called “process sensors,” which “derive process data from the raw data [] of the executed processes” and generate event logs “from the derived process data.” EX1004, [0072], [0080]; *id.*, [0091] (“[A] process sensor is a unit which can be applied to the raw data of the source system to sense one or multiple process steps.”). When executed, process sensor units apply transformations that extract information

relating to the process that generated the raw data, such as element identifier, step identifier, and timestamp. *Id.*, [0090]-[0102].

The process sensor units may be “scripted in a programming language” or defined “through a graphical user interface” (*id.*, [0115]-[0117]). From these inputs, “executable code for the underlying execution platform is generated.” *Id.*, [0119]; EX1003, ¶44.

Rinke937 discloses multiple “source systems” (*id.*, [0090], [0103]) and its process sensor units sense and extract data that is “often scattered among different tables” (*id.*, [0096]), which might require “consecutive join[s] over (possibly) multiple tables” to derive the event log information (*id.*, [0099]). In one example, Rinke937’s process sensor unit retrieves data from an invoice table BSEG, and process-step and timestamp data from change log tables CDPOS and CDHDR, respectively. *Id.*, [0135]. The process sensor unit relates those three different tables and transforms the retrieved data into a common event log. *Id.*, [0130]-[0135]; EX1003, ¶45.

B. Nooijen (EX1005)

Nooijen, titled “Automatic Discovery of Data-Centric and Artifact-Centric Processes,” was published in 2013 by Springer-Verlag. EX1005, 316. Nooijen qualifies as prior art under Section 102(a)(1). *Id.*, (i)-(v) (2013 Copyright and catalog information); *Amazon.com, Inc. v. Vocalife LLC*, IPR2021-01331, Paper 11 at 29-

30 (Jan. 31, 2022) (finding Springer-Verlag an established publisher); EX1015, 872 (citing Nooijen); EX1003, ¶46.

Like Rinke937, Nooijen relates to transformation of raw process data (stored in relational data tables) into an event log that can be used for process mining. EX1005, 316-319. Nooijen seeks to improve existing approaches through techniques for “automatically discovering a process model” (*id.*, 316-317), including “automatic discovery of schema-log mapping needed for log extraction” (*id.*, 326). Nooijen explains that this avoids a manually-defined “mapping between source data and event log.” *Id.*, 318; EX1003, ¶¶47-48.

Nooijen’s “automated approach” automatically reconstructs undocumented relationships among source-data tables, identifies a main table and the related tables for a process object, determines the event specification for that object, and generates and executes database queries that transform the related table data into a timestamp-ordered event log for process mining analysis. EX1005, 317; *id.*, 318-324. Nooijen Fig. 1 provides an overview:

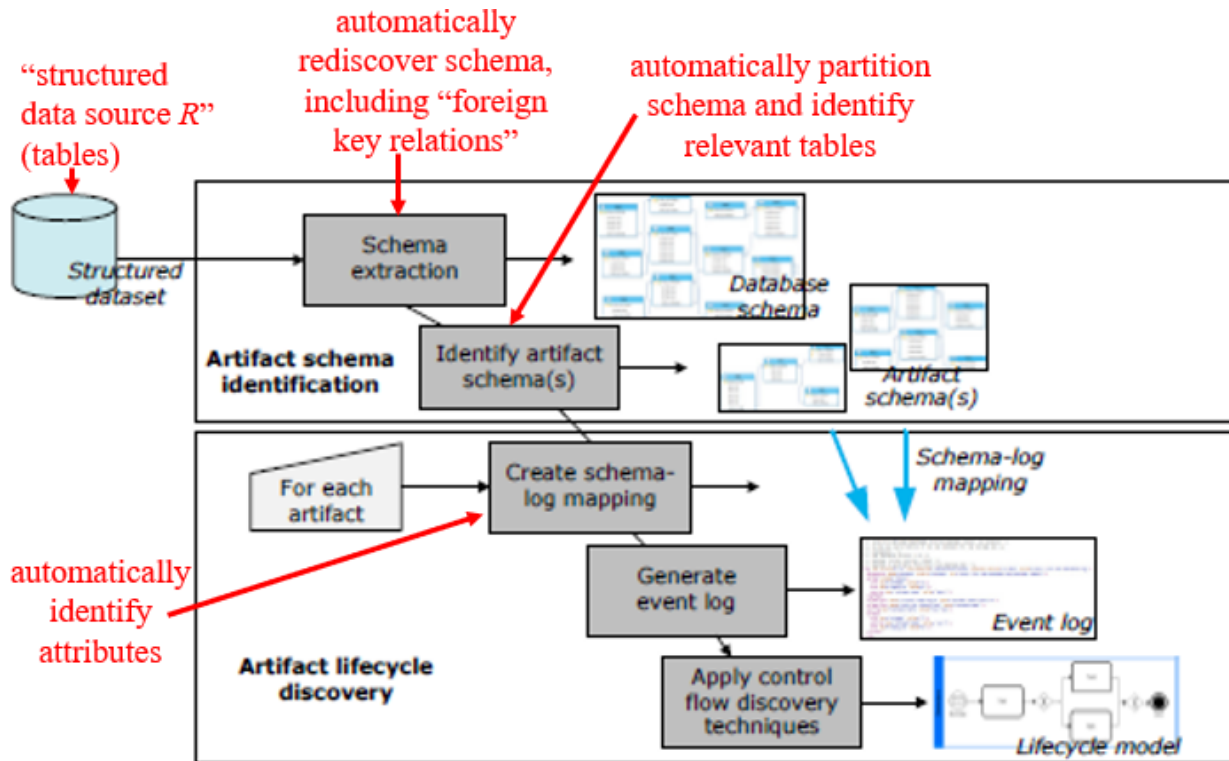


Fig. 1. Approach to Artifact Discovery

Id., 317 (annotated in red); EX1003, ¶¶47, 49.

For a given data source, the Nooijen technique determines and reconstructs a relational schema S that represents the source tables, including column types, primary keys, and foreign key relationships. EX1005, 319. After rediscovering and reconstructing the schema, Nooijen partitions the schema into artifact schemas. *Id.*, 317. Each artifact schema describes the information model for one artifact (business entity) and includes the tables containing information relevant to that artifact. *Id.*; EX1003, ¶¶50-51.

Nooijen also identifies a main table, T^m , for each artifact schema. EX1005, 320. The main table's primary key serves as the identifier of the artifact, while other tables provide additional attributes for that artifact. *Id.*, 319-321. Nooijen then automatically extracts an event specification, referred to as a schema-log mapping, from the artifact schema, using the selected main table. That event specification is used to construct database queries that extract events and their attributes, group the events into cases, order them by timestamp, and write the result into a classical log. *Id.*, 321-324; EX1003, ¶51.

Nooijen adapts “the log extraction technique of XESame,” with an algorithm that “fully automatically” creates an “event log-trace mapping” that would otherwise be specified manually. EX1005, 324. With that mapping, “XESame automatically generates the database queries needed to extract the log.” *Id.* Those queries extract events, informed by the reconstructed foreign-key relationships, and the resulting event traces are grouped by trace identifier, ordered by timestamp, and written to an XES format event log. *Id.*; EX1003, ¶52.

C. Harrison (EX1006)

U.S. Patent Application Publication No. 2018/0096000 to Harrison et al. (“Harrison”) was published on April 5, 2018, and qualifies as prior art under 35 U.S.C. § 102(a)(1). EX1006, Cover; EX1003, ¶53.

Like Nooijen, Harrison is directed to identifying relationships between tables. Harrison discloses techniques for consolidating information stored in separate computer systems. It explains that related information may be stored in different source systems—for example, separate business units may maintain different databases containing information about the same customers. EX1006, [0065]. Because organizations need “to bring together data from diverse data sources to enable processing and analysis of data sets” (*id.*, [0003]), Harrison teaches a system for importing entire data tables from multiple structured sources, such as relational databases, and storing copies of those tables in a central data repository. *Id.*, [0053]-[0055]; EX1003, ¶54.

Harrison’s “Data Tap” tool imports data “from any number of data sources” into a central repository referred to as a “data lake.” EX1006, [0055]-[0056]; *see also id.*, [0059] (describing importing tables “from multiple data sources”). External data sources may include Oracle relational databases or other database-management systems. *Id.*, [0058]. Harrison states that the tool “may import a database in its entirety (i.e. including all tables).” *Id.*, [0059]; EX1003, ¶55. “[T]ypically all columns of the source table will be imported.” *Id.*, [0061], [0089]-[0090]. Harrison explains that this centralized architecture permits an organization to “bring together information from many disparate databases ... and analyse [sic] and process the data

centrally,” rather than requiring separate analysis on each external source system. *Id.*, [0064]-[0066]; EX1003, ¶¶56-57.

After importing and saving the data tables locally, Harrison teaches to analyze them to “automatically discover relationships” between the tables, including primary- and foreign-key relationships. EX1006, [0067]; *see also id.*, [0068], [0221]-[0225]. These relationships can be used to construct join queries that, when executed, combine and extract data from the tables. *Id.*, [0068], [0235]; *see also id.*, [0418]-[0424]; EX1003, ¶58.

Harrison is analogous art, as arising in the same field of endeavor (gathering distributed data and preparing it for analysis) and being reasonably pertinent to the problem purportedly solved by the '708 patent. The problem involves identifying relationships among data tables originating from different computer systems or processing units, so that their data can be analyzed. *E.g.*, EX1001, 2:3-58, 12:39-13:17. Harrison directly addresses that problem with teachings on importing disparate data collections created by different operations, discovering relationships among the different source data, and using the identified relationships to facilitate analysis, e.g., through queries and joins. *E.g.*, EX1006, [0064]-[0068], [0410], claim 22. A POSITA seeking to solve the problem purportedly addressed by the '708 patent would therefore logically consult Harrison's relationship-discovery techniques. EX1003, ¶59.

D. Combining Rinke937, Nooijen, And Harrison

A POSITA would have been motivated, with a reasonable expectation of success, to implement an event log generation system as disclosed by Rinke937 with Nooijen's automated relationship discovery teachings and Harrison's centralized import-and-analyze teachings. Such a system would reduce manual effort and efficiently generate event logs from process data created on multiple source systems. EX1003, ¶60.

The resulting Rinke937-Nooijen-Harrison system would include the following operations, which can be grouped into two stages:

A first stage that creates an executable process-sensor unit using information derived from locally imported data tables:

- Import data tables from different source computer systems and store them in local storage, as taught by Harrison (EX1006, [0055]-[0063], [0078]-[0090]), the data tables being generated by processes running on those source computer systems, as taught by Harrison (*e.g., id.*, [0064] (import from disparate databases “supporting different operations of the organization”), [0065] (“importing data” from systems for gas-supply and maintenance processes));
- Analyze the imported tables to determine and reconstruct relationships between tables, including undocumented primary-key and foreign-key

relationships, as taught by Harrison (*id.*, [0067]-[0068], [0221]-[0225]), using the automated schema discovery taught by Nooijen (EX1005, 317, 319-20);

- Select, from the imported tables, a “main table” whose records identify the individual business objects for which an event log will be created, as taught by Nooijen (*id.*, 317, 319-22);
- Select related event tables based on the main table and reconstructed relationships, and determine activities associated with each identified object, as taught by Nooijen (*id.*, 320-24);
- Create an executable transformation using the selected tables, reconstructed relationships, and determined activities, by combining Nooijen’s teachings on automatic query generation and event log creation (*e.g.*, *id.*, 324) with Rinke937’s teachings on event log generation (EX1004, [0031]-[0032], [0096]-[0102], [0116]-[0119]);

A second stage that executes the executable process-sensor unit to generate an event log from “raw data stored in the source systems” (EX1004, [0090]):

- Execute the transformation to extract object identifiers and associated activity data from the tables on the different source systems, as taught by Rinke937 (*id.*, [0012]-[0020], [0090]-[0102], [0112]-[0119]) and Nooijen (EX1005, 324), using network connection information as

taught by Harrison (EX1006, [0055], [0087]-[0090], [0455]-[0456]);
and

- Store the extracted data as an event log, associating each object with its process steps and their order, and provide that log to a process mining system for analysis and reconstruction, as taught by Rinke937 (EX1004, [0017]-[0026], [0035], [0075]-[0088], [0112]-[0113]).

EX1003, ¶60.

POSITAs would make routine implementation choices when embodying this combination and would have made this combination for at least the following reasons. EX1003, ¶61.

First, a POSITA would have been motivated to use the automated discovery and mapping teachings of Nooijen to reduce the manual effort of creating the executable process-sensor units taught by Rinke937. Rinke937 explains that relevant data fields are “scattered among different tables” and that relationships “have to be defined within the process sensor(s),” which requires manual definition of table relationships through a GUI or programming/scripting. EX1004, [0096]; *id.*, [0097]-[0102], [0115]-[0119]. Nooijen recognizes the downsides of “manually” defining such mappings, and teaches creating the mapping “fully automatically.” EX1005, 324. Nooijen explains that manually selecting tables and defining source-to-log mappings is “hard and time consuming,” suffers from human bias, and is often

“incomplete.” *Id.*, 317-319. Nooijen addresses these issues with its automated approach that reconstructs relationships, identifies tables relevant to an artifact/object, and generates the queries needed to extract related events. *Id.*, 323-24, 326. A POSITA thus would have naturally applied Nooijen’s teachings to automate the determination of tables, relationships, and other information when creating the process sensors taught by Rinke937, to reduce manual effort and improve accuracy. *See id.*, 318; EX1003, ¶62.

Second, a POSITA would have been motivated to use Harrison’s techniques for importing data tables locally to improve Nooijen’s automated determination of tables, relationships, and other information. Harrison identifies a “need to bring together data from diverse data sources to enable processing and analysis,” explaining that such data is often “interrelated.” EX1006, [0003], [0064]-[0068]. A POSITA knew that this was often the case for event logs / process protocols, as taught by Nooijen and Rinke937. *E.g.*, EX1010, 12 (describing the need to merge data “distributed over a variety of sources”), 15. And like Nooijen, Harrison notes that relationships “may be lost during the import process” and that relationships between tables from different data sources “may not be known a priori,” and teaches automatically discovering relationships among imported tables. EX1006, [0066]-[0068], [0421]; *id.*, [0005], [0055], [0059]-[0065]; *see also* EX1005, 317, 319. Moreover, Nooijen’s automated reconstruction of relationships involved substantial

processing (EX1005, 319-20, 325), which POSITAs knew was more reliably and efficiently performed on locally imported data rather than disparately located data. EX1003, ¶¶63-65; *see also* EX1008, 316. A POSITA would have thus naturally combined Harrison’s complementary central import functionality with the Rinke937-Nooijen teachings, to more effectively determine table relationships and create process sensor units in those common situations where process data is generated on different source systems. EX1006, [0055], [0059], [0064]-[0065]; EX1010, 12, 15; EX1003, ¶66.

Third, Rinke937 teaches that its process sensor units operate on “raw data stored in the source systems” to create event logs. EX1004, [0090]; *see also id.*, [0091]-[0102], [0117]-[0119]. POSITAs understood that this allows the executable process sensor unit to be reused and repeatedly executed to generate event logs as additional activities are recorded in the source systems, as taught by Rinke937. *Id.*, [0063], [0090], [0113]. Rinke937 expressly identifies “advantages” of “using process sensors to create event logs” including configuration “to specifics of the source data system” and execution “at fixed time intervals.” *Id.*, [0056]-[0064], [0113], [0090]-[0102], [0117]-[0119]. Thus, at least for the repeated execution of a process sensor unit as taught by Rinke937, a POSITA would have been motivated to execute the process sensor units to generate event logs from “raw data stored in the source systems.” *Id.*, [0090]; EX1003, ¶67.

Finally, a POSITA would have had a reasonable expectation of success combining these teachings because Rinke937, Nooijen, and Harrison describe compatible operations and information. Nooijen describes successful use of its automated technique to improve a system much like that of Rinke937, namely one which transforms source data into event logs for process mining. EX1004, [0010], [0072], [0080], [0090]-[0102]; EX1005, 316-24. Both references teach using table relationships as part of extracting data fields for inclusion in an event log. EX1004, [0031]-[0032], [0095], [0103]-[0119]; EX1005, 321-24. These references thus provide complementary teachings, and a POSITA would have expected success combining them into a system in which Nooijen's automation teachings supply the table relationships and other mapping information used by Rinke937 to create executable process-sensor code and generate event logs. Similarly, Harrison expressly contemplates use of its centralized import teachings in systems that determined primary-key and foreign-key relationships between disparate data sources. EX1006, [0058]-[0068], [0221]-[0235]. Nooijen uses the same kinds of tables and relationships to generate its mappings, and Rinke937 uses table relationships to define the extraction it performs. EX1005, 317, 319-24; EX1004, [0096]-[0102]. A POSITA would have thus seen Harrison's teachings as a natural and readily implemented extension of the Rinke937-Nooijen event log generation process. POSITAs would have recognized that the combination involved the

predictable application of known database, software, and data-mining techniques according to their established functions. EX1003, ¶68; *infra* Section VII.E.8.

So combined, Rinke937-Nooijen-Harrison taught and rendered obvious claims 1-14, as described in detail below. EX1003, ¶69.

E. Claim 1

1. Element [1.1]

Rinke937-Nooijen-Harrison teaches the preamble. EX1003, ¶70.

a) Element [1.1.a]

[1.1.a] A computer-implemented method of creating a process protocol in a local computer system (LC), which comprises a processor and a storage device, wherein the process protocol encodes object related activities and is created from raw data, which is stored in part in a first external computer system (EC1) and in part in a second external computer system (EC2),

Rinke937-Nooijen-Harrison satisfies this element. As discussed below, Rinke937-Nooijen-Harrison teaches creating, in a local computer system, a process protocol encoding object-related activities from raw data stored in part in two external computer systems. EX1003, ¶71.

Rinke937 expressly discloses “a method for generating, in a computer system having a processor and a storage means operatively coupled to the processor, an event log from raw data stored in a source system” (*a computer-implemented method for creating a process protocol in a local computer system (LC), which comprises a processor and a storage device, wherein the process protocol encodes object related*

activities and is created from raw data). EX1004, [0012]; *id.*, [0001], [0051]-[0055]. Rinke937-Nooijen-Harrison includes the computer system taught by Rinke937 and generates, in that computer system, an event log from raw data as taught by Rinke937. *Supra* Section VII.D; EX1003, ¶72.

The event log of Rinke937-Nooijen-Harrison is a *process protocol*. The “event log” taught by Rinke937 has a “data structure comprising at least a first attribute for storing the unique identifier of the process element of the respective process step, a second attribute for storing the identifier of the respective process step, and a third attribute for storing the order of the process steps within a process element.” EX1004, [0020]-[0023]; *id.*, [0014]-[0019] (disclosing a process sensor unit deriving the unique identifier(s), identifier(s) and order of steps from the raw data), [0005]-[0006], [0025]-[0026], [0075]-[0088], [0091]-[0102], [0124]-[0136], Fig. 5. Likewise, the “data structure of the process protocol” described by the ’708 patent “comprises at least a first attribute, in which a unique identifier of the process of the respective process step is stored, a second attribute, in which an identifier of the respective process step is stored, and a third attribute, in which a sequence of the process steps within a process is stored.” EX1001, 9:8-15; *see also id.*, 15:21-49; EX1003, ¶73.

Rinke937 “shows an example of an event log” with this data structure:

Rinke937 “Event Log”

Unique ID	Process Step	Timestamp (Order)
1	Purchase Requisition	2012-01-01 13:05
1	Purchase Requisition Approval	2012-01-01 15:09
1	Purchase Order Item	2012-01-02 17:03
1	Goods Receipt	2012-01-03 13:01
1	Invoice Receipt	2012-01-03 15:00
2	Purchase Requisition	2012-01-05 11:00
2	Purchase Requisition Approval	2012-01-05 17:00
...		

EX1004, [0086] (red annotations added). The ’708 patent provides an example “process protocol” with similar attributes as the event log:

’708 Patent “Process Protocol”

Identifier of the process	Identifier of the process step	Sequence
1	Change Price	01.01.2018
1	Change Amount	02.01.2018
1	Change Amount	03.01.2018
1	Change Price	04.01.2018
1	Change Price	05.01.2018

EX1001, 15:42-48 (red annotations added). The “event log” taught by Rinke937 is therefore a *process protocol* because its data structure includes the same attributes and information regarding a process as the “process protocol” described by the ’708 patent. EX1003, ¶74; *see also* EX1002, 196/228.

The event log of Rinke937-Nooijen-Harrison encodes object related activities. Specifically, the event log taught by Rinke937 records “an unique ID, a

process step, and an order of the process steps,” where the unique identifier (ID) corresponds to a “process object.” EX1004, [0081]-[0084], [0099]. Each event log entry thus associates a process step with an identified “object,” and the event log therefore *encodes object related activities. E.g., id.*, [0124]-[0129] (providing example where the objects are invoices); EX1003, ¶75.

The event log of Rinke937-Nooijen-Harrison is *created from raw data which is stored in part in a first external computer system and in part in a second external computer system.* Specifically, as explained in the following paragraphs, Rinke937-Nooijen-Harrison creates an event log from “raw data stored in the source systems” as taught by Rinke937, where that raw data is stored in part in a first external computer system and in part in a second external computer system as taught by Harrison. EX1004, [0090]; EX1006, [0055], [0058]-[0059], [0065], [0456]; EX1003, ¶76.

As noted above, Rinke937 discloses creating the event log (*process protocol*) “based on raw data stored in a source system, i.e., to transform the source data into an event log format for further process mining analysis.” EX1004, [0010]; *see also id.*, [0008], [0035], [0079], [0088], [0114], [0121]. Its process sensor units “derive process data from the raw data comprising the digital traces of the executed processes” and generate the event log from that data. *Id.*, [0080]; *see also id.*, [0001], [0003], [0008], [0012]-[0023], [0072]-[0079], [0090]-[0094]. The source data is

stored in database tables, and the relevant fields may be “scattered among different tables.” *Id.*, [0096]; *see also id.*, [0097]-[0102], [0124]-[0136]. Rinke⁹³⁷ further discloses raw data in plural “source systems.” *Id.*, [0090], [0103]. EX1003, ¶77.

Harrison discloses importing data “from any number of data sources 102-1, 102-2, 102-3 into a data repository 108,” including “tables and data” originally stored in “multiple data sources.” EX1006, [0055], [0059]. Harrison depicts these data sources as separate and distinct in Figure 1:

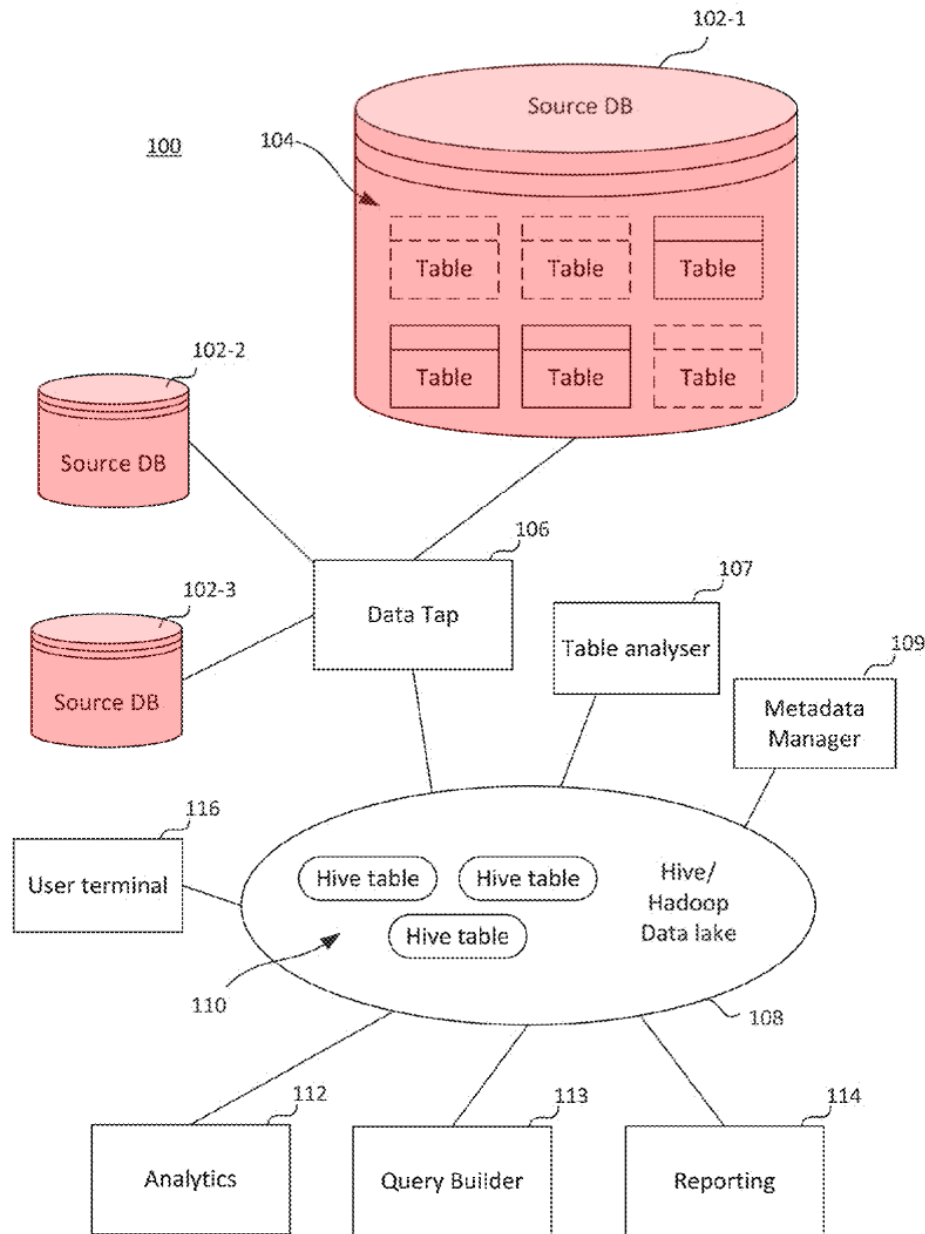


Fig. 1

EX1006, FIG. 1 (red annotations added); EX1003, ¶78.

Harrison further describes that “it may often be the case that data imported from disparate data sources is interrelated,” with an example of “a gas or similar utilities provider” that “import[s] a database of gas supply accounts from a supply

part of the organisation and a database of boiler maintenance data from a service/maintenance part of the organisation,” while “similar relationships may occur between disparate data sources maintained by organisations within any field (e.g. medical, banking, manufacturing etc.).” EX1006, [0065]. Harrison also contemplates heterogeneous source databases using different database management platforms. *Id.*, [0058]. Consistent with that distributed architecture, Harrison further discloses that its data-management server includes a “network interface . . . for connection to data sources 102” over a network such as the Internet. *Id.*, [0456]; Fig. 26. A POSITA would have understood from these disclosures that “disparate data sources” 102-1, 102-2, and 102-3 are implemented on different computer systems external to the local computer system at least because these data sources are separately maintained, accessible via a network, and comprise different databases. *Id.*, [0055], [0059], [0078]-[0090], [0454]-[0456], FIG. 26; EX1003, ¶79. For example, in Rinke937-Nooijen-Harrison, the computer system hosting data source 102-1 (a *first external computer system*) and the separate computer system hosting data source 102-2 (a *second external computer system*) store respective portions of the raw process data used to create the event log. EX1006, [0055], [0059], [0064]-[0065], [0456], Fig. 1; EX1003, ¶¶76-80.

The combination of Harrison’s “disparate data sources” teachings with the event log generation framework of Rinke937, which contemplates multiple “source

systems,” results in a local computer system creating an event log from raw data stored partly in one external computer system and partly in another. EX1004, [0012]-[0023], [0090], [0112]-[0113]; EX1006, [0055], [0059], [0064]-[0065], [0087], [0454]-[0457]. Confirming the obviousness of such an implementation, the ’708 patent’s Background acknowledges that related processes may be executed and recorded on different computer systems. EX1001, 1:43-67; EX1003, ¶80.

The Rinke937-Nooijen-Harrison computer system is a *local computer system*. As noted above in this section, the Rinke937-Nooijen-Harrison combination includes a “computer system having a processor” that generates “an event log from raw data stored” in multiple “source systems.” *E.g.*, EX1004, [0012]; *id.*, [0013]-[0023], [0051]-[0055], [0184]-[0188]. The computer system is a *local computer system* at least because it is “different” from the source systems (*external systems*). *See* EX1001, 4:49-53 (“Hereby, the terms ‘external’ and ‘internal’ only indicate that different computer systems are concerned.”); EX1003, ¶81.

b) Element [1.1.b]

[1.1.b] wherein the raw data in the external computer systems is stored in a number of data tables,

As explained for element [1.1.a], Rinke937 teaches multiple sources and storing “raw data” in multiple data tables, and in Rinke937-Nooijen-Harrison source data is stored in data tables on the different source computer systems as taught by

Harrison. EX1004, [0090]-[0102], [0124]-[0136]; EX1006, [0055], [0058]-[0059]; *supra* Section VII.E.1.a; EX1003, ¶82.

Rinke937 expressly states that “[i]n a source system which handles processes the data is usually saved into *tables* which reside in a data storage, e.g. in a database system.” EX1004, [0096] (emphasis added). Further, “[t]he different data fields which are sensed by the process sensor(s) are often scattered among different tables.” *Id.* Those fields contain the raw data used to create the event log: Rinke937’s process sensor units operate on “source data, i.e., raw data stored in the source systems” to identify process-step information. *Id.*, [0090]; *see also id.*, [0012]-[0023], [0075]-[0080], [0091]-[0102]. Thus, as Rinke937-Nooijen-Harrison includes the arrangement of multiple external computer systems as data sources taught by Harrison, Rinke937-Nooijen-Harrison stores the raw data used to create the process protocol in multiple data tables within the first and second external computer systems, thus teaching [1.1.b]. EX1004, [0090], [0096]-[0102], [0124]-[0136]; EX1006, [0055], [0058]-[0059]; EX1003, ¶¶83-85.

c) Element [1.1.c]

[1.1.c] wherein the raw data comprises data, which is created during the execution of processes in the first external computer system and in the second external computer system,

Rinke937-Nooijen-Harrison satisfies this element. Rinke937 teaches that its raw source system data comprises records generated during process execution, and

as explained above for element [1.1.a], Rinke937-Nooijen-Harrison includes multiple external source systems, as taught by Harrison, that provide such process-generated data from both the first and second external computer systems. EX1004, [0003]-[0008], [0075]-[0080], [0090]-[0096], [0124]-[0136]; EX1006, [0055], [0058]-[0065]; *supra* Section VII.E.1.a; EX1003, ¶86.

Raw data created during the execution of processes: Rinke937 explains that tasks reflecting process steps “are conducted in an IT driven environment and leave traces in an IT system,” and that “[p]icking up these traces and reconstructing the as-is process from this data is the goal of process mining.” EX1004, [0003]. Rinke937 accordingly teaches “deriv[ing] process data from the raw data comprising the digital traces of the executed processes.” *Id.*, [0080]; *id.*, [0008]. Those traces reside in the “raw data stored in the source systems.” *Id.*, [0090]; *id.*, [0103]. The process is reconstructed “solely from the traces which have been left on the source system.” *Id.*, [0075]. Rinke937 further explains that a process sensor unit derives process steps from the “raw data stored in the source systems” and that “[t]he respective changes which were triggered in the raw data are then converted into the aforementioned structure of an event log.” *Id.*, [0090]-[0091]; *see also id.*, [0072], [0076]-[0079], [0090]-[0095]. Rinke937 provides concrete examples confirming that these records arise during execution of operational processes on the source systems. *Id.*, [0124]-[0136]; *id.*, [0090], [0103]; EX1003, ¶¶87-88.

Process-generated data in the first and second external computer systems: Harrison teaches consolidating data from “many disparate databases,” including databases that “support[] different operations of the organisation,” because “it may often be the case that data imported from disparate data sources is interrelated.” EX1006, [0064]-[0065]. One example includes “a database of gas supply accounts from a supply part of the organisation” and “a database of boiler maintenance data from a service/maintenance part of the organisation.” *Id.*, [0065]. These databases contain related records because “some supply customers may also be maintenance customers.” *Id.* A POSITA would have understood this example as teaching related processes executing on multiple external computer systems, including at least a first external computer system and a second external computer system, because it was well known that raw process data was often generated and stored across multiple different source systems. *Supra* Section IV.B.2. Harrison also expressly contemplates “changes in the data sources” as records in the data sources are created and updated over time. EX1006, [0205]; *id.*, [0063]; EX1003, ¶89.

Thus, in Rinke937-Nooijen-Harrison, the raw data used to create the event log includes data generated during execution of processes in each of the first and second external computer systems. EX1004, [0075]-[0080], [0090]-[0096], [0124]-[0136]; EX1006, [0064]-[0065]; EX1003, ¶90.

d) Element [1.1.d]

[1.1.d] wherein a process comprises several process steps, wherein each process step creates the data during execution, and wherein the method comprises:

Rinke937-Nooijen-Harrison satisfies this element. In Rinke937-Nooijen-Harrison, “[b]usiness processes consist of multiple process steps” (*process comprises several process steps*) with “raw data comprising the digital traces of the executed processes” (*each process step creates the data during execution*), where the execution of the processes creates the “raw data stored in the source systems,” as taught by Rinke937. EX1004, [0080], [0090], [0120]; *id.*, [0003]-[0008], [0075]-[0079], [0081]-[0088], [0091]-[0102], [0112]-[0113], [0124]-[0136], Figs. 2, 5; *supra* Section VII.E.1.c; EX1003, ¶¶91-95.

2. Element [1.2]

[1.2] (a) importing the data tables from the first external computer system and from the second external computer system into the local computer system, and storing the imported data tables in the storing device of the local computer system,

Rinke937-Nooijen-Harrison satisfies this element. As discussed above, Rinke937-Nooijen-Harrison *imports the data tables from the first external computer system and from the second external computer system into the local computer system. Supra* Section VII.D. Rinke937-Nooijen-Harrison “import[s] data from any number of data sources 102-1, 102-2, 102-3 into a data repository 108,” as taught by

Harrison. EX1006, [0055], [0059], Fig. 1; *supra* Sections VII.D, VII.E.1.a. Those data sources include separate relational databases, with each source database containing multiple data tables. EX1006, [0058]-[0059]; *see also id.*, [0003]. Rinke937-Nooijen-Harrison imports “a database in its entirety (i.e. including all tables)” or “may import only one or more selected tables” from the databases stored in respective data source systems, as taught by Harrison. *Id.*, [0059]; *see also id.*, [0061], [0089], [0159]-[0164]; EX1003, ¶96.

Rinke937-Nooijen-Harrison stores the imported table data in the storage device of its computer system (*storing the imported data tables in the storing device of the local computer system*) so that the computer system can “analyse and process the data centrally,” as taught by Harrison. EX1006, [0064]. As explained above, Rinke937-Nooijen-Harrison combines the table-import functionality of Harrison with the computer system (*local computer system*) taught by Rinke937, which includes “a storage means being operatively coupled to the processor.” EX1004, [0052]-[0055]; *supra* Sections VII.D, VII.E.1.a. Harrison teaches importing source tables into a data repository using “any data storage technology,” including “a conventional structured database such as a relational database.” EX1006, [0055]-[0056], [0059]. The computer system storing the imported data also performs “processing functions,” including analysis of the imported data. *Id.*, [0461]. Harrison

thus teaches storing imported tables within the same computer system that processes them. EX1003, ¶97.

Rinke937-Nooijen-Harrison thus stores the imported table copies taught by Harrison in the local storage taught by Rinke937, for analysis using Nooijen’s automated techniques. *Supra* Section VII.D. This arrangement makes the table contents locally available for repeated examination and comparison when reconstructing table relationships, without repeatedly retrieving those contents from the external source systems. *See* EX1005, 319-20, 324-25. The imported tables therefore reside in the storage device of the local computer system, while the original tables remain in the external computer systems. EX1003, ¶98.

3. Element [1.3]

[1.3] (b) in the local computer system, determining and reconstructing the relationships between the imported data tables,
--

Rinke937-Nooijen-Harrison satisfies this element by determining relationships between the locally imported data tables and reconstructing a schema representing those relationships, including previously unknown relationships between tables imported from different computer systems. *Supra* Sections VII.D, VII.E.2; EX1003, ¶99.

Nooijen expressly teaches that “first the schema S_R including column types and primary and foreign keys is rediscovered,” including “non-documented foreign

key relations.” EX1005, 317. Nooijen’s schema extraction techniques “rediscover foreign keys between tables” and identify “undocumented foreign key relations (used in the application, but not declared as such).” *Id.*, 319. The result is a relational schema containing the table schemas and the identified foreign keys. *Id.* Figure 1 of Nooijen depicts this workflow, with “Schema extraction” operating on a “Structured dataset” to produce a “Database schema”:

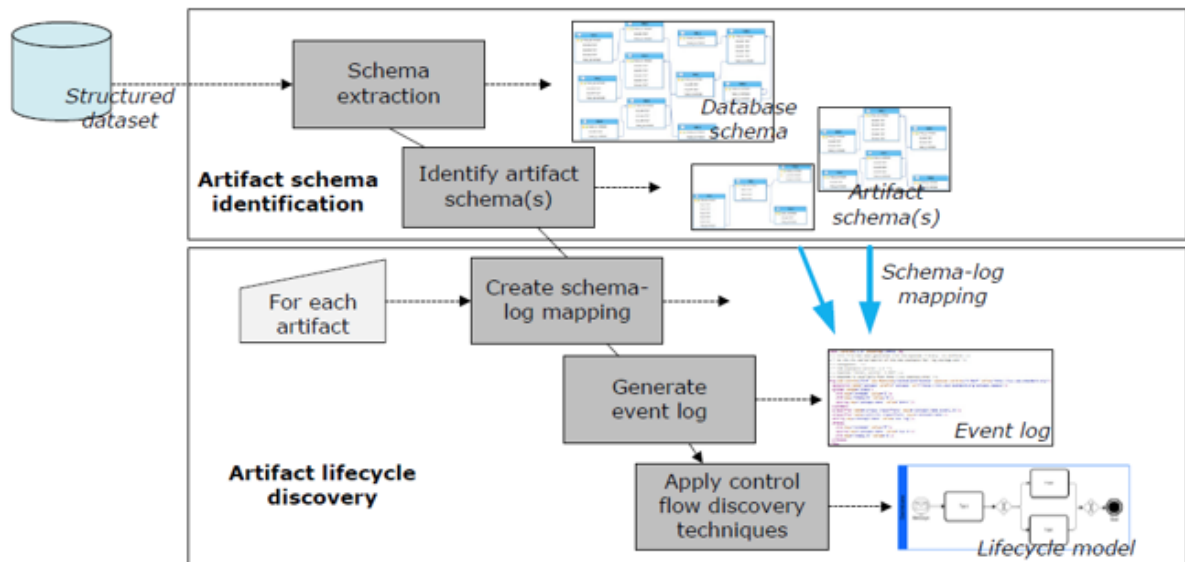


Fig. 1. Approach to Artifact Discovery

EX1005, Fig. 1; *id.*, 317; EX1003, ¶¶99-100.

The “foreign key relations” in Nooijen are *relationships between data tables*. Each foreign key connects a parent table and its primary-key columns with a child table and its referencing columns. EX1005, 319. Identifying that connection determines which columns relate records in one table to records in another, and

including the connection in the reconstructed schema represents the relationship between those tables. *Id.* This is consistent with the '708 patent, which identifies reconstructed relationships as “foreign key relationships.” EX1001, 12:10-13. Harrison similarly describes identifying “a primary key to foreign key relationship,” meaning a relationship between a primary key in one table and a corresponding foreign key in another. EX1006, [0221]; EX1003, ¶101.

Rinke937-Nooijen-Harrison performs this determining and reconstructing, as taught by Nooijen, on the imported tables stored in the local computer system after importing the tables into the local storage in Rinke937 for analysis. *Supra* Sections VII.D, VII.E.2. Nooijen calls its input a “relational database *R*,” such as “an ERP system’s database.” EX1005, 319. In Rinke937-Nooijen-Harrison, that input is the local relational repository containing the imported copies of the raw source-system tables described in Rinke937 and Harrison. EX1004, [0090], [0096]; EX1006, [0056], [0059]; *supra* Sections VII.E.1.b, VII.E.2; EX1003, ¶102.

The relationships determined in Rinke937-Nooijen-Harrison include previously unknown relationships between tables imported from different computer systems. Harrison expressly observes that relationships between tables that “originated from different data sources ... may not be known a priori.” EX1006, [0421]. Harrison describes separately maintained gas-supply and boiler-maintenance databases containing overlapping customer information. EX1006,

[0065]. Related columns need not have matching names, and source documentation may be “incomplete or inconsistent.” *Id.*, [0066]. Such cross-system relationships need not have been identified when the individual source databases were designed. Zhang, a background reference cited by Nooijen, expressly explains that database designers may omit foreign-key constraints because “they are not aware of implicit relationships inherent in the data” and where “such relationships might hold across multiple databases.” EX1009, 805; *see* EX1005, 319, 327 (reference [21]). Zhang describes discovery over a collection of tables potentially originating in multiple databases without requiring prior knowledge of the data. EX1009, 806. Thus, automated foreign-key discovery was understood to identify relationships that were previously unknown, including relationships spanning different databases. EX1003, ¶¶103-106.

A POSITA would have applied the automated schema-discovery techniques taught by Nooijen to the combined collection of locally imported data tables to identify cross-system relationships that had not been previously identified or supplied through a predefined cross-system data model. *Supra* Section VII.D; EX1003, ¶¶106-107. Harrison expressly motivates this analysis by teaching automated discovery of relationships from imported table contents, including overlapping column values, for use in creating join queries. EX1006, [0067]-[0068], [0221]-[0225]. Applying the automated discovery techniques taught by Nooijen in

that setting determines the previously unknown cross-system relationships after import and represents them in the reconstructed schema. EX1005, 317, 319; EX1003, ¶107.

These reconstructed table relationships are used in creating the process sensor units taught by Rinke937. Rinke937 explains that relevant fields are “scattered among different tables” and that “[t]he relations between the different tables have to be defined within the process sensor(s).” EX1004, [0096]. Those relationships permit joins between tables containing object identifiers, process-step information, and ordering attributes. *Id.*, [0099]-[0102]. Rinke937 illustrates this with a process sensor unit that, when executed, retrieves the invoice identifier from BSEG, process-step information from CDPOS, and the timestamp from CDHDR, explaining that “these three tables have to be related within this process sensor.” *Id.*, [0135]. In Rinke937-Nooijen-Harrison, the foreign-key relationships reconstructed as taught by Nooijen identify the columns connecting those related tables and supply the relationships needed to define the sensor joins. EX1005, 319; *supra* Section VII.D; EX1003, ¶¶108-109.

4. Element [1.4]

[1.4] (c) in the local computer system, determining and selecting a first data table from the number of imported data tables, wherein each data record stored in the first data table represents an object for which the process protocol is to be created,

Rinke937-Nooijen-Harrison satisfies this element. Nooijen identifies “a unique main table . . . with the least distance to all other tables in the cluster” and designates that table as the main table of an artifact schema (*determining and selecting a first data table*). EX1005, 320. In Rinke937-Nooijen-Harrison, this identification and selection occur in the local computer system and operate on the locally imported tables described for elements [1.2] and [1.3]. *Supra* Sections VII.D, VII.E.2-VII.E.3; EX1003, ¶110.

Rinke937-Nooijen-Harrison determines and selects a first data table. The “main table” in Nooijen is a table identifying the business objects whose activities are to be recorded in the event log. EX1005, 316, 319-320. Nooijen explains that “[t]he idea of artifacts is to decompose this data into the business objects, or artifacts, of the process.” *Id.*, 319. An artifact schema describes the information of one artifact and “consists of all tables that contain relevant information” about that artifact. *Id.*, 317. Nooijen identifies these schemas “automatically through clustering,” grouping tables based on their contents and foreign-key relationships, and identifying the main table within each cluster (*determining and selecting a first data table from the number of imported data tables*). *Id.*, 320; EX1003, ¶111.

Each record represents an object for which the process protocol is to be created. Nooijen explains that each “artifact instance has a unique identifier” and that “the primary key of the main table is the identifier of the artifact.” EX1005, 320.

Each record of the main table, distinguished by its primary-key value, therefore represents an artifact instance (*object*). EX1003, ¶112. Nooijen discloses joining event information with the main table, “thus associating each event with exactly one trace.” EX1005, 324. Thus, in Rinke937-Nooijen-Harrison, each record of the main table (*first data table*) identifies an object whose activities are recorded in an event log, as taught by Nooijen. *See id.*, 317; *see also* EX1004, [0081]-[0084], [0099]; *supra* Sections VII.D, VII.E.1.a; EX1003, ¶¶112-114.

5. Element [1.5]

[1.5] (d) in the local computer system, determining and selecting a number of second data tables from the number of imported data tables, which are necessary for the creation of the process protocol, based on the selected first data table and based on the reconstructed relationships between the imported data tables, wherein each data record stored in the number of second data tables represents a process step,

Rinke937-Nooijen-Harrison satisfies this element by applying the automated mapping techniques taught by Nooijen to the locally imported tables and reconstructed relationships described above. *Supra* Sections VII.D, VII.E.2, VII.E.4. Nooijen teaches using the selected main table to identify related event tables (*second data tables*) supplying the process-step information for the objects represented in that main table. EX1005, 321-24; EX1003, ¶115.

Rinke937-Nooijen-Harrison determines and selects second data tables based on the selected first data table and the reconstructed relationships.

Nooijen identifies schema-to-log mappings through an automated approach “based on timestamps and foreign key relations,” using the reconstructed relationships. EX1005, 321; *see also id.*, 317, 319-20; *supra* Section VII.E.3. Nooijen identifies event types by “selecting all columns with a timestamp domain as event columns, except for columns that are part of a parent table of the main table,” and identifies the event table containing each selected column. EX1005, 322-323. Selection thus depends on the main table (*selected first data table*) and the reconstructed parent-child relationships between tables; the selected event tables T_{ET} containing those event columns, other than the main table, are the imported source tables selected as *second data tables*. *Id.*; EX1003, ¶116.

Each event table is linked to the selected main table. Algorithm 1 passes the main table and a given event table to the CreateMapping algorithm, which determines their connecting path through Algorithm 2. EX1005, 322. That path “consists of the sequence of foreign keys connecting those tables,” linking the event information to the corresponding objects. *Id.*, 323. In Rinke937-Nooijen-Harrison, these relationships are used to define joins in the process sensor units taught by Rinke937. EX1004, [0096]-[0102]; *supra* Section VII.D; EX1003, ¶117.

Each record in the selected second data tables represents a process step. Nooijen teaches that a given “event table T_{ET} ” contains an event column C_e that includes the time stamp of each event. EX1005, 321-323 (“ T_{ET} is the table that

contains C_e ” with “event column C_e defining the time stamp of the event”). A POSITA would have understood that each record (row) of the table thus represents a process step that occurred at the time indicated by the timestamp. Nooijen confirms this, teaching that “[e]ach record in the joined table defines an event with the given name, that occurred at the time-stamp written in [the column] C_e .” EX1005, 324; *see also id.*, 322-324 (explaining Algorithms 1-2). Similarly, Rinke937 describes “the ‘change date’ of the respective field as timestamp of the process step” and, in its example tables, describes “the timestamp from the table ‘CDHDR.’” EX1004, [0132], [0135]. In Rinke937’s example, a change-of-content sensor extracts recorded invoice changes from source-system change logs, using the changed field to identify the process step and the change date as its timestamp. EX1004, [0130]-[0135]. Thus, in Rinke937-Nooijen-Harrison, each record in the event tables (*second data tables*) corresponds to a process step that occurred at a time indicated by a timestamp. EX1003, ¶118.

The selected second data tables supply information necessary to create the process protocol. Nooijen discloses joining the event tables with the main table through the identified foreign-key relationships; “[e]ach record in the joined table defines an event with the given name” and includes the trace-identifier, “thus associating each event with exactly one trace.” EX1005, 324. These event identifiers and timestamps supply the process-step information grouped by object identifier and

ordered by timestamp to populate the event log (*process protocol*). *Id.* Applying this selection to the locally imported tables and reconstructed schema selects the necessary second data tables in the local computer system. *Supra* Sections VII.D, VII.E.2-VII.E.4; EX1003, ¶¶119-120.

6. **Element [1.6]**

[1.6] (e) in the local computer system, determining activities for the objects of the selected first data table based on the selected second data tables and the reconstructed relationships between the selected first data table and the selected second data tables,

Rinke937-Nooijen-Harrison satisfies this element by applying the automated techniques taught by Nooijen to the locally imported tables and reconstructed relationships. *Supra* Sections VII.D, VII.E.2-VII.E.5; EX1003, ¶121.

Rinke937-Nooijen-Harrison determines activities for the objects of the selected first data table. Nooijen explains that “all event types in the artifact schema are identified automatically.” EX1005, 321. For each event type, Algorithm 1 creates an event mapping containing an event name, event-identifier columns, timestamp column, and links to the relevant tables; specifically, “the event column C_e and an event name are added” to the mapping. *Id.*, 321-323. The named event types identify the activities, while the event identifiers and timestamps identify particular occurrences of those activities. These correspond to the process-step descriptions and ordering attributes used by Rinke937 to generate entries for the event log,

including named activities such as “Invoice paid.” EX1004, [0092]-[0094], [0100]-[0101], [0124]-[0129]; EX1003, ¶122.

Rinke937-Nooijen-Harrison determines the activities based on the selected second data tables and reconstructed relationships and associates the determined activities with the objects in the first data table. Nooijen identifies event information through an approach “based on timestamps and foreign key relations,” with Algorithms 1 and 2 connecting each event table to the main table through a path that “consists of the sequence of foreign keys connecting those tables.” EX1005, 321-23; *supra* Sections VII.E.3, VII.E.5. The event mappings specify joins, including intermediate tables where needed, through which “[e]ach record in the joined table defines an event with the given name,” and includes the trace identifier from the main table, “thus associating each event with exactly one trace.” EX1005, 324. Those trace identifiers identify the objects represented by the records of the main table. *Supra* Section VII.E.4. The mappings therefore determine which activities relate to each identified object, supplying the object identifiers, process-step descriptions, and ordering information used by the process sensor units in Rinke937. EX1004, [0099]-[0102]; *supra* Section VII.D; *infra* Section VII.E.7; EX1003, ¶¶123-125.

7. **Element [1.7]**

[1.7] (f) in the local computer system, creating an executable transformation, which based on

[i] the selected first data table,

[ii] the selected second data tables, and

[iii] the relationships between the selected first data table and the selected second data tables

[iv] is adapted to extract data from the first data table and from the second data tables based on the determined activities, and to store the extracted data as process protocol, and

Rinke937-Nooijen-Harrison satisfies this element. EX1003, ¶126.

Rinke937-Nooijen-Harrison creates a process sensor unit (*executable transformation*) in its computer system (*local computer system*). Rinke937 teaches “creation of the process sensors” from “relations, source tables, target tables and other attributes” and generating “executable code” from those definitions. EX1004, [0116]-[0119]; *see also id.*, [0012], [0027], [0075], [0098], [0110]-[0114], claims 1, 4, 7-10. That code implements transformations “to transform the source data into an event log format for further process mining analysis,” including transformations using fields from joined tables. *Id.*, [0010], [0072], [0096]-[0102]. The resulting executable process sensor units extract process data from the source systems and create the event log (*process protocol*). *Id.*, [0090]-[0091], [0113], [0120]-[0121]; *see also id.*, [0027], [0056]-[0063]. Rinke937 expressly states that

“the processor converts the independent representation of the sensor statement into an executable representation,” with executable code generated “during runtime.” *Id.*, [0031], [0117], [0119], claims 1, 8. A POSITA therefore would have implemented this code generation in the combined local computer system described above. *Supra* Sections VII.D, VII.E.2-VII.E.6; EX1001, 4:49-53 (explaining that “external” and “internal” indicate different computer systems); EX1003, ¶¶127-129. Nooijen confirms that its automatically generated mapping supplies input from which “XESame automatically generates the database queries needed to extract the log,” supporting use of the same table, field, and relationship information to generate executable process sensor units in Rinke937. EX1005, 324; EX1003, ¶130.

Rinke937-Nooijen-Harrison creates the process sensor unit (*executable transformation*) based on the main table (*selected first table*), event tables (*selected second tables*), and the foreign key relationships between them (*relationships between the selected first data table and the selected second data tables*) (sub-elements [i]-[iii]). In Rinke937-Nooijen-Harrison, the automated mappings from Nooijen supply the sensor-definition information: the main-table identifier columns identify the fields used to derive the “unique identifier” corresponding to the “process object”; the event mappings specify the “process step description” and “order attribute,” such as a timestamp; and the reconstructed foreign-key relationships define the “consecutive join over (possibly) multiple

tables” used to retrieve that information. EX1004, [0099]-[0101]; EX1005, 321-24. The mappings identify the main table (*selected first data table*), related event tables (*selected second data tables*), and foreign-key relations connecting them (*relationships between the selected first and second data tables*), as explained above. *Id.*, 321-23; *supra* Sections VII.E.3-VII.E.5. Those inputs supply the source tables, fields, and joins scripted into the process sensor definitions from which an executable process sensor unit is generated, as taught by Rinke937. EX1004, [0096]-[0102], [0117]-[0119]; EX1003, ¶¶131-133.

Rinke937-Nooijen-Harrison process sensor units (*executable transformations*) extract data from the selected tables based on the determined activities and store that data as an event log (*process protocol*) (sub-element [iv]). Rinke937 teaches that its process sensor units derive “process data” by extracting the data for particular process steps across different tables, and converting the resulting information into event-log entries. EX1004, [0080], [0090]-[0094], [0096]-[0102], [0120]-[0121]. The process sensor units then store “the generated data as event log with the storage means” (*store the extracted data as a process protocol*). *Id.*, [0013]-[0023]; *supra* Section VII.E.1.a (element [1.1.a]). Each Nooijen event mapping specifies an event name (*determined activity*), event identifier, timestamp, and table relationships identifying how to retrieve the associated information. EX1005, 321-23; *supra* Section VII.E.6. Incorporated into

the sensor definitions, that information directs extraction of the object identifiers and associated process-step data from the selected tables. EX1004, [0099]-[0102], [0119]-[0121]; EX1005, 321-24. The resulting extraction and storage thus implement the determined activity mappings. EX1004, [0097]-[0102], [0119]-[0121]; *infra* Section VII.E.8; EX1003, ¶¶134-137.

8. Element [1.8]

[1.8] (g) in the local computer system, executing the transformation, wherein during execution of the transformation the data is extracted from the data tables stored in the external computer systems (ES1, ES2), and storing the extracted data as process protocol, and providing the created process protocol to a Process Mining System for analysis and reconstruction of the processes carried out in the first external computer system and in the second external computer system.

Rinke937-Nooijen-Harrison satisfies this element. EX1003, ¶138.

The Rinke937-Nooijen-Harrison computer system (*local computer system*) executes the process sensor unit (*the transformation*) to extract data from the tables stored in the source systems (*external computer systems*). Rinke937 describes the processor “executing ... at least one process sensor” to “derive process data from the [source systems’] raw data,” generate data from that derived raw data, and store the generated data as an “event log.” EX1004, [0012]-[0020]; *see also id.*, [0010], [0021]-[0023]. The process sensor unit extracts data from “raw data stored in the source systems” (*data extracted from the data tables stored in the external computer systems*), using table definitions and joins to retrieve

the process-step information, and may execute “in a scheduled manner,” including daily. *Id.*, [0090], [0096]-[0102], [0103], [0113]. Once the mappings identify the relevant tables, fields, and relationships, subsequent executions retrieve the mapped data without repeating the broader relationship-discovery analysis. Reading the original source tables captures newly recorded activities without requiring a separate refresh of the entire locally imported table collection. EX1003, ¶¶139-141.

Harrison provides the metadata connecting the locally analyzed tables to their external sources. Its import metadata identifies the source system or database, table name, original column name, and any renamed column in the imported copy. EX1006, [0111]-[0117]. A POSITA would have used that metadata to translate the locally determined table and column references in the Nooijen mappings into corresponding external-source references when generating the process sensor unit taught by Rinke937. EX1005, 321-324; EX1004, [0117]-[0119]. In Rinke937-Nooijen-Harrison, the locally created executable process sensor unit implements the locally determined mappings while reading data from the corresponding original tables in the first and second external computer systems, as explained in the combination section. *Supra* Sections VII.D, VII.E.1.a, VII.E.7. Where related tables reside in different external systems, the local Rinke937-Nooijen-Harrison computer system retrieves the required fields separately and joins the retrieved records using the key-column correspondences specified in the mappings. Matching an object

identifier from the main table with the corresponding foreign-key value from an event table associates the event with its object; intermediate tables supply additional matching columns where needed. EX1003, ¶¶140-143.

Rinke937-Nooijen-Harrison stores the extracted data as an event log (*process protocol*). Rinke937 expressly teaches “storing, by the processor, the generated data as event log with the storage means.” EX1004, [0020]. The stored records associate object identifiers with process-step identifiers and ordering information, including timestamps. *Id.*, [0021]-[0026], [0075]-[0085], [0097]-[0102]. Rinke937-Nooijen-Harrison retains these operations to store the activity data extracted from both external systems as the event log (*process protocol*) described above. *Supra* Sections VII.D, VII.E.1.a; EX1003, ¶144.

Rinke937-Nooijen-Harrison provides the created process protocol to a Process Mining System for analysis and reconstruction of the processes carried out in the external computer systems. Rinke937 states that “[t]he generated data stored with the storage means may be provided to a process mining system,” and that, “[g]iven such an event log, the as-is process may be reconstructed by process mining algorithms.” EX1004, [0035], [0088]; *see also id.*, [0001], [0008], [0010], [0039], [0071]-[0080], [0120]-[0122], [0137]-[0140]. Because the combined event log records the process steps carried out in both external systems, providing that log enables analysis and reconstruction of those processes by the Process Mining

System. *Supra* Sections VII.D, VII.E.1.a, VII.E.1.d. Thus, Rinke937-Nooijen-Harrison teaches each element of claim 1 and their claimed combination and renders claim 1 obvious. EX1003, ¶¶145-146.

F. Claim 2

[2] The method of claim 1, wherein at least one of the selected data tables is imported from the first external computer system (ES1), and at least one of the selected data tables is imported from the second external computer system (ES2).

Rinke937-Nooijen-Harrison selects at least one data table imported from each external computer system. Rinke937-Nooijen-Harrison imports tables from multiple systems and applies reconstructed relationships to select the main table and related event tables, as explained above. *Supra* Sections VII.D, VII.E.2-VII.E.5. Harrison teaches importing tables “from any number of data sources” and that “it may often be the case that data imported from disparate data sources is interrelated.” EX1006, [0055], [0058]-[0059], [0065], Fig. 1; *see also id.*, [0003], [0005]-[0006]. Applying the timestamp and foreign-key relationship selection techniques taught by Nooijen to related process data distributed across those systems would result in the main table (*first data table*) and one or more of the event tables (*second data tables*) coming from “disparate” but “interrelated” data sources. *Id.*, [0055]; EX1005, 321-23; EX1003, ¶¶147-148.

For example, Harrison describes importing gas-supply and boiler-maintenance databases from different parts of an organization, explaining that “some

supply customers may also be maintenance customers.” EX1006, [0065]. Applying Rinke937-Nooijen-Harrison to analyze activities for those shared customers would have made it obvious to select a customer-identifying main table from the gas-supply system (*at least one selected data table imported from the first external computer system*) and a related event table recording maintenance activities from the service/maintenance system (*at least one selected data table imported from the second external computer system*). The reconstructed relationships would associate the maintenance events with the corresponding customers, consistent with Nooijen’s teaching of joining event information with the main table to associate each event with a trace. EX1005, 324; EX1003, ¶149.

A POSITA would have been motivated to select tables from both systems to include relevant activities for the shared customers in the resulting event log that selection from only one system would otherwise omit. This predictably applies the table-selection and relationship-mapping techniques taught by Nooijen and renders claim 2 obvious. EX1003, ¶150.

G. Claim 3

[3] The method of claim 1, wherein during reconstructing of the relationships between the imported data tables in the step (b), it is determined via which attributes the data tables are related to each other.

Rinke937-Nooijen-Harrison determines the attributes through which the imported data tables are related during reconstruction of their relationships.

Supra Sections VII.D, VII.E.3. Nooijen teaches reconstructing a schema containing foreign-key relationships, each defined by the parent table and its primary-key columns (*attributes*), together with the child table and its referencing columns (*attributes*). EX1005, 319. Discovering these relationships, including “undocumented foreign key relations,” thus identifies the particular attributes linking the records in the related tables. *Id.*; EX1003, ¶¶151-152.

Harrison likewise identifies “relationships between table columns” based on whether the columns contain keys and the overlap between their data contents; its “Table Analyser” examines imported table contents to identify a relationship between a primary key in one table and a corresponding foreign key in another. EX1006, [0068], [0221]. In Rinke937-Nooijen-Harrison, these identified columns supply the relationships used to define the sensor joins taught by Rinke937. EX1004, [0096]-[0102]; *supra* Section VII.D. Accordingly, Rinke937-Nooijen-Harrison renders claim 3 obvious. EX1003, ¶¶152-153.

H. Claim 4

[4] The method of claim 3, wherein two data tables are related to each other, if an attribute of the one data table is identical or similar to an attribute of the other data table.
--

Rinke937-Nooijen-Harrison identifies a relationship between two tables having identical customer identifier attributes. Harrison identifies “a relationship between the CUSTID column of table 2302 and the CUSTID column of table 2304”

as the strongest proposed relationship between those tables, which “may have originated from different data sources.” EX1006, [0421], Fig. 23A. The CUSTID column of table 2302 (*attribute of one data table*) and the CUSTID column of table 2304 (*identical attribute of the other data table*) provide the relationship between the two tables, thus satisfying the claimed “identical or similar” alternative. EX1003, ¶¶154-155.

Harrison identifies relationships by considering whether columns are keys and the degree of overlap between their values, including an example in which customer-identifier columns serve as a primary key in a customers table and a foreign key in an orders table. EX1006, [0068], [0231]-[0233]. The relationship therefore rests on shared customer identifier attributes and corresponding key data, rather than relying solely on matching column labels. EX1003, ¶156.

Rinke937-Nooijen-Harrison represents these relationships through the primary key and referencing columns identified by Nooijen and uses them for the table joins taught by Harrison and Rinke937, as explained above. EX1005, 319; EX1006, [0235], [0422]; EX1004, [0099]-[0102]; *supra* Sections VII.D, VII.E.3, VII.G. Accordingly, Rinke937-Nooijen-Harrison renders claim 4 obvious. EX1003, ¶157.

I. Claim 5

[5] The method of claim 1, wherein the relationships reconstructed between two imported data tables are stored, wherein the names of the two data tables, the respective attributes, and an assignment of the attributes of the one data table to the attributes of the other data table are stored.

Rinke937-Nooijen-Harrison stores the reconstructed relationships, including the related table names, attributes, and assignments between attributes. Nooijen represents each reconstructed foreign key as $F=(T_p, C_p, T_c, C_c)$, identifying the parent table T_p and child table T_c (*the two related imported data tables*), the primary key columns C_p and referencing columns C_c (*the respective attributes*), and the foreign key relation linking those columns (*the assignment between the attributes*). EX1005, 319. Harrison expressly stores “information on identified table relationships” in “an output file 906,” including table and column identifiers and entries for related column pairs. EX1006, [0234], [0289]-[0296]; EX1003, ¶¶158-159.

A POSITA would have used the table and column names in this stored representation to identify and join the correct tables through the corresponding attributes. Harrison describes “named columns” and permits “any appropriate encoding or referencing” to identify the source table and column. EX1006, [0240], [0245]. Nooijen uses the reconstructed schema to generate mappings containing the main table and links to other tables “in the form of foreign keys” for constructing

database queries. EX1005, 321. Storing the table names, column names, and their pairings would predictably preserve the discovered relationships for those extraction operations in Rinke937-Nooijen-Harrison. EX1003, ¶160.

In Rinke937-Nooijen-Harrison, this stored relationship information provides the relationships used to generate and execute the process sensor units, as explained above. *Supra* Sections VII.D, VII.E.3, VII.E.7. Rinke937 scripts the “relations, source tables, target tables and other attributes” in the sensor definitions and generates executable code from those definitions, and maintains a data model describing the structure of the elements and additional tables linked to the event log. EX1004, [0099], [0117]-[0119]. The stored table names, attributes, and attribute assignments thus support the joins performed by the process sensor units when extracting the process data. *Id.*, [0099]-[0102]. Accordingly, Rinke937-Nooijen-Harrison teaches and renders claim 5 obvious. EX1003, ¶161.

J. Claim 6

[6] The method of claim 1, wherein a first data model belonging to the data tables imported from the first external computer system (EC1) is imported from the first external computer system into the local computer system, wherein the relationships between the data tables imported from the first external computer system (EC1) are reconstructed based on the first data model, a second data model belonging to the data tables imported from the second external computer system (EC2) is imported from the second external computer system into the local computer system, wherein the relationships between the data tables imported from the second external computer system (EC2) are reconstructed based on the second data model, such that only the relationships between the data tables from the first external computer system and the data tables from the second

external computer system have to be reconstructed without the use of a data model.

Rinke937-Nooijen-Harrison imports a data model from each external computer system into the local computer system. Harrison teaches extracting table metadata from source databases and storing their schema information in a metadata repository. EX1006, [0087], [0103], [0111]-[0125], Fig. 2B. This includes “relationship information ... extracted from the source database.” *Id.*, [0382]. Applying those teachings to the two external systems imports the schema metadata and relationship information from the first source database (*first data model*) and the corresponding information from the second source database (*second data model*) into the Rinke937-Nooijen-Harrison computer system (*local computer system*). EX1003, ¶¶162-163.

Rinke937-Nooijen-Harrison uses each imported data model to reconstruct relationships among the tables imported from the corresponding external system. Nooijen represents foreign-key relationships through the parent and child tables and their respective primary-key and referencing columns, identifying the tables and attributes to be linked. EX1005, 319. Harrison accommodates both relationships extracted from a source database and relationships discovered from imported table contents. EX1006, [0382]-[0383]. In Rinke937-Nooijen-Harrison, the imported schema and relationship definitions from each

source system are used to establish a local representation of the relationships among its imported tables: the first source schema (*first data model*) provides the relationships among tables from the first external system, and the second source schema (*second data model*) provides the relationships among tables from the second external system. EX1003, ¶164.

Rinke937-Nooijen-Harrison uses the imported models for internal relationships so that only relationships between tables from different external systems require reconstruction without a data model. Nooijen explains that additional relationship discovery “may be required” when schema information is incomplete and discovers “undocumented foreign key relations.” EX1005, 319. A POSITA would have used the imported models for documented relationships and applied discovery as taught by Nooijen to missing relationships. For source systems whose models document their internal table relationships, this leaves relationships between tables across the two systems to be reconstructed without a source model. Like Nooijen, Harrison supports that analysis by identifying relationships between imported tables, including tables that “originated from different data sources.” EX1006, [0068], [0221], [0421]. This implementation of Rinke937-Nooijen-Harrison avoids unnecessary rediscovery of documented relationships, consistent with Harrison limiting analysis to new relationships rather than fully analyzing all possible column combinations. EX1006, [0314]. The resulting combined schema

provides the relationships used by the automated mappings in Nooijen and the process sensor units in Rinke937, as explained above, thus rendering claim 6 obvious. *Supra* Sections VII.D, VII.E.3, VII.E.7; EX1003, ¶¶165-166.

K. Claim 7

[7] The method of claim 1, wherein during determining and selecting of a first data table in the step (c), it is determined, which data table meets one or more conditions, wherein the conditions are selected from a group, consisting of the data table comprises less data sets compared to the other data tables, the data table comprises an attribute, in which the time stamps are stored, the data table is referenced by a few other data tables, and combinations thereof.

As explained for claim 1, Rinke937-Nooijen-Harrison imports the source tables, reconstructs their relationships, and applies the automated selection in Nooijen to identify a main table (*selected first data table*) whose records identify the objects for which the event log (*process protocol*) will be created. *Supra* Sections VII.D, VII.E.2-VII.E.4. Claim 7 permits selection of the first data table based on any one of the listed conditions. Rinke937-Nooijen-Harrison determines and selects a main table using foreign-key relationships with related auxiliary tables, as taught by Nooijen, thus teaching the condition that *the data table is referenced by a few other data tables*. EX1003, ¶167.

Rinke937-Nooijen-Harrison uses foreign key references from related auxiliary tables to identify the main table. Nooijen explains that business objects materialize as “important” tables, while their additional attributes materialize in

related “auxiliary” tables. EX1005, 320. Its selection technique groups these tables into artifact schemas using a “distance function between tables” that considers table “importance and foreign key relations.” *Id.* Specifically, “[f]oreign key relations between tables associate auxiliary tables to important tables.” *Id.* Nooijen defines those relations through a parent table and its primary key columns, together with a child table and its referencing columns. *Id.*, 319. Thus, auxiliary child tables containing foreign keys reference the important parent table through its primary key. Those foreign key references contribute to the distances used to cluster the tables and select “a unique main table” having “the least distance to all other tables in the cluster.” *Id.*, 320. Accordingly, the foreign key references are used in determining and selecting the main table. EX1003, ¶168.

Rinke937-Nooijen-Harrison selects a main table that is referenced by a few related auxiliary tables. The technique in Nooijen selects the main table from a group of related tables. EX1005, 320. The related auxiliary tables reference the main table through the foreign-key relationships considered in the selection analysis (*the data table is referenced by a few other data tables*). *Id.* The evaluated artifact schema A2 in Nooijen contains six tables, including the main table, illustrating application of the relationship-based selection technique to a small group of related tables. *Id.*, 325. The technique selects the main table using distances reflecting table importance and foreign-key relationships with the relevant auxiliary tables. *Id.*, 320,

325. In Rinke937-Nooijen-Harrison, the system performs this analysis on the locally imported tables using the reconstructed relationships, as explained for claim 1. *Supra* Sections VII.D, VII.E.3-VII.E.4. Accordingly, Rinke937-Nooijen-Harrison renders claim 7 obvious. EX1003, ¶¶169-170.

L. Claim 8

[8] The method of claim 7, wherein information with respect to the selected first data table is stored.

Rinke937-Nooijen-Harrison stores information with respect to the main table (*selected first data table*). *See supra* Section VII.K. Nooijen teaches recording information regarding the main table in the generated schema-log mapping, including the main table, its trace identifier columns, foreign-key relations to other tables, and associated attribute mappings (*information with respect to the selected first table*). EX1005, 320-22. The trace mapping, which includes the specification of the main table, is stored for subsequent query generation and event extraction. *Id.*, 324; EX1003, ¶¶171-172.

Rinke937 likewise teaches “storing auxiliary data” and maintaining a data model describing the tables and relationships used to create the event log. EX1004, [0036]-[0039]. In Rinke937-Nooijen-Harrison, the mapping information taught by Nooijen is stored, as taught by Rinke937, to provide the definitions for the source table and relationships to the process sensor framework taught by Rinke937. *Id.*,

[0036]-[0039], [0117]-[0119]; *supra* Section VII.D. Accordingly, Rinke937-Nooijen-Harrison renders claim 8 obvious. EX1003, ¶172.

M. Claim 9

[9] The method of claim 1, wherein during determining and selecting of the number of second data tables in the step (d) those data tables are selected, which meet one or more conditions, wherein the conditions are selected from the group, consisting of the data tables comprise many data sets compared to the first data table, the data table is referenced directly or indirectly by the first data table, the data tables respectively comprise an attribute, in which the time stamps are stored, and combinations thereof.

As explained for claim 1, Rinke937-Nooijen-Harrison applies the automated mapping techniques taught by Nooijen to select related event tables (*second data tables*) from the locally imported tables based on the main table (*first data table*) and the reconstructed relationships. *Supra* Sections VII.D, VII.E, VII.E.5; EX1003, ¶173.

Rinke937-Nooijen-Harrison selects second data tables that are referenced directly or indirectly by the first data table. Nooijen uses foreign-key relationships to associate auxiliary tables with the main table. EX1005, 319-320. For each selected event table T_{ET} (*second data table*), the mapping algorithm determines foreign keys $F_{link} = \text{Path}(T^m, T_{ET})$, linking the main table T^m (*first data table*) to that event table directly or through intervening tables. *Id.*, 321-322; *id.*, 323. The algorithm verifies foreign keys “in the child direction and the parent direction,” thus identifying related event records from the main table even where the foreign-key

attribute resides in the event table. *Id.*, 323. The “path between two tables [] consists of the sequence of foreign keys connecting those tables”: one relationship provides a direct connection, while a sequence through intervening tables provides an indirect connection. *Id.*, 323. Nooijen uses those relationships to join the event table with the main table, including intervening tables where needed, and associate the extracted events with the corresponding object. *Id.*, 324. These connections satisfy the claimed condition, consistent with the usage in the ’708 patent. *See* EX1001, 12:20-22, 13:6-14, 14:1-12. Thus, Rinke937-Nooijen-Harrison selects event tables linked to the main table through foreign key paths (*data tables referenced directly or indirectly by the first data table*), as taught by Nooijen. EX1003, ¶¶174-177.

Rinke937-Nooijen-Harrison also selects second data tables based on those tables containing attributes in which timestamps are stored. Nooijen identifies event types by “selecting all columns with a timestamp domain as event columns” and defines the corresponding event table (T_{ET}) as the table containing that timestamp column. EX1005, 322-323. For each selected event column C_e (*attribute in which timestamps are stored*), Nooijen identifies the event table T_{ET} (*selected second data table*) containing that column. *Id.*, 322-23. Thus, the timestamp attribute is an additional criterion used to identify and select the event table, teaching the condition *the data tables respectively comprise an attribute, in which the time stamps are stored*. EX1003, ¶178.

Rinke937-Nooijen-Harrison therefore teaches the additional element recited in claim 9 and renders claim 9 obvious. EX1003, ¶179.

N. Claim 10

[10] The method of claim 9, wherein information with respect to the selected second data tables and with respect to the relationships between the selected second data tables and the first data table are stored.

Rinke937-Nooijen-Harrison stores information regarding the selected second data tables and their relationships with the selected first data table. Nooijen records the event identifier columns and timestamp column (*information regarding each selected second data table*), the main table (*selected first data table*), and the foreign key relations connecting those tables (*relationships between the selected second data tables and the first data table*) in its generated event mappings. EX1005, 321-322. As explained for claim 9, these paths connect each selected event table to the main table either directly or through intervening tables. *Id.*, 322-323; *supra* Section VII.M. These mappings are retained and supplied to the extraction tool to generate the database queries that join the identified tables using the recorded relationships. EX1005, 324. In Rinke937-Nooijen-Harrison, this stored information provides the table and relationship definitions used to generate the process sensor units in Rinke937, as explained for element [1.7]. EX1004, [0102]; *supra* Sections VII.D, VII.E.7. Accordingly, Rinke937-Nooijen-Harrison teaches the additional element recited in claim 10 and renders claim 10 obvious. EX1003, ¶¶180-181.

O. **Claim 11**

[11] The method of claim 1, wherein during creating of the executable transformation in the step (f) it is determined, which attributes of the selected first data table and of the selected second data tables are considered during extracting of the data.

Rinke937-Nooijen-Harrison determines which attributes of the selected first and second data tables will be considered during extraction when creating the executable transformation. Nooijen generates mappings identifying the trace identifier columns of the main table (*attributes of the selected first data table*) and the event identifier and timestamp columns of the event tables (*attributes of the selected second data tables*). EX1005, 321-22. It also assigns additional columns as instance or event attributes and creates attribute mappings specifying each attribute name, type, and source column C_a (*attribute considered during extraction*). *Id.*, 321-323. These mappings therefore identify the particular columns to use before the extraction queries are generated. *Id.*, 324; EX1003, ¶¶182-183.

In Rinke937-Nooijen-Harrison, those column determinations supply the field definitions used to create the process sensor unit (*executable transformation*), as explained for element [1.7]. *Supra* Sections VII.D, VII.E.7. Rinke937 expressly defines the source-table fields supplying the object identifier, process-step information, timestamp, and additional event-log attributes, and generates executable code from the scripted or graphical sensor definitions. EX1004, [0099]-

[0102]; *id.*, [0117]-[0119]. The attributes identified through the mappings in Nooijen are incorporated into the transformation during its creation, thus teaching the additional element of claim 11 and rendering this claim obvious. EX1003, ¶184.

P. Claim 12

[12] The method of claim 11, wherein during executing of the transformation in the step (g) data sets are read out from the first data table and from the second data tables, wherein the data sets, which have been read out, comprise the attribute values of the attributes determined for the respective data table, wherein the data sets, which have been read out, are stored as process protocol, wherein each data set stored in the process protocol represents a process step (Event) of a process executed in the first external computer system and in the second external computer system.

Rinke937-Nooijen-Harrison reads data sets containing the determined attribute values from the selected first and second data tables during execution of the process sensor unit (*transformation*). As explained for claim 11, the automated mappings in Nooijen provide the table and attribute definitions incorporated into the process sensor unit. *Supra* Section VII.O. Rinke937 teaches that the sensor definition “is evaluated during the execution of the sensor,” retrieving values from the defined fields through joins across the source tables. EX1004, [0096]-[0102]. The process sensor units thus read the object identifier fields from the main table (*selected first data table*) and the process step, timestamp, and additional attribute fields from the related tables (*selected second data tables*), as specified during creation of the executable process sensor unit. In Rinke937-

Nooijen-Harrison, the process sensor unit reads the raw data in the external source systems, as explained for element [1.8]. *Supra* Section VII.E.8; EX1003, ¶¶185-186.

Rinke937-Nooijen-Harrison further stores the resulting data sets read out from the first and second data tables as an event log (*process protocol*), with each stored record representing a process step. Rinke937 expressly teaches storing the data generated during process sensor unit execution as an event log. EX1004, [0014]-[0023]. Rinke937 further specifies that the identifier of a single process step, the associated object identifier, and the ordering information form “a single data record” (*data set representing a process step*). *Id.*, [0025]. The process sensor units of Rinke937-Nooijen-Harrison contribute records reflecting the process steps carried out across the first and second external systems to the complete event log, as explained for claim 1. *Supra* Sections VII.E.1, VII.E.8. Accordingly, Rinke937-Nooijen-Harrison teaches the additional element of claim 12 and renders this claim obvious. EX1003, ¶187.

Q. Claim 13

[13] The method of claim 12, wherein the data sets, which have been read out, are stored according to a predetermined data structure in the process protocol, wherein the predetermined data structure comprises at least: a first attribute, in which a first unique identifier of the process of the respective process step is stored, a second attribute, in which an identifier of the respective process step is stored; and a third attribute, in which a sequence of the process steps within a process is stored.

As discussed above with regard to element [1.1.a], Rinke937 discloses the same data structure recited in claim 13. *Supra* Section VII.E.1.a. Specifically, Rinke937 expressly discloses that the event log (*process protocol*) has a “predetermined data structure comprising at least a first attribute for storing the unique identifier of the process element of the respective process step, a second attribute for storing the identifier of the respective process step, and a third attribute for storing the order of the process steps within a process element.” EX1004, [0020]-[0023]; *id.*, [0025]-[0026], [0081]-[0086]. Rinke937-Nooijen-Harrison uses this “predetermined data structure” taught by Rinke937 for the event log when storing the records generated by its process sensor units, as explained for claim 12. *Supra* Section VII.P. Accordingly, Rinke937-Nooijen-Harrison teaches the additional element of claim 13 and renders this claim obvious. EX1003, ¶¶188-189.

R. Claim 14

[14] The method of claim 13, wherein the attribute values of the attributes of the process protocol are determined from the attribute values of the data sets, which have been read out, and/or the table names and/or column names of the selected first data table and/or the selected second data tables.

Rinke937-Nooijen-Harrison determines the attribute values of the attributes of the event log from the attribute values read from the selected data tables. Rinke937 teaches deriving the object identifier, process step information, timestamp, and “further attributes” for the event log (*attribute values of the process*

protocol) from the fields of the source table, including through transformations and joins across related tables. EX1004, [0099]-[0102]. As a concrete example, the “Change of content sensor” taught by Rinke937 retrieves the “unique identifier” from the invoice table BSEG, the “process step” from the change log table CDPOS, and the “timestamp” from the change log table CDHDR, and stores the resulting values in the event log. *Id.*, [0130]-[0136]; EX1003, ¶¶190-191.

In Rinke937-Nooijen-Harrison, the mappings in Nooijen identify the fields used by the process sensor units, as explained for claims 11-13. *Supra* Sections VII.O-VII.Q. Accordingly, Rinke937-Nooijen-Harrison teaches the “attribute values” alternative recited in claim 14, and renders claim 14 obvious. EX1003, ¶192.

VIII. NO OBJECTIVE INDICIA OF NON-OBVIOUSNESS

Petitioner is not aware of any evidence of objective indicia of non-obviousness having a nexus to the challenged claims.

IX. CONCLUSION

Petitioner requests institution for claims 1-14 on Ground 1 specified in this petition.

Respectfully submitted,

Dated: October 23, 2026

By: /Andrew M. Mason/
Andrew M. Mason, Reg. No. 64,034
andrew.mason@klarquist.com

Counsel for Petitioner

**CERTIFICATE OF COMPLIANCE WITH
TYPE-VOLUME LIMITATION PURSUANT TO 37 C.F.R. § 42.24**

This brief complies with the type-volume limitation of 37 C.F.R. § 42.24(a)(1)(i).

The brief contains **12,660** words, excluding the parts of the brief exempted by 37 C.F.R. § 42.24(a).

The brief has been prepared in a proportionally spaced typeface using Microsoft Word for O365 in a 14-point Times New Roman font.

Dated: October 23, 2026

By: /Andrew M. Mason/

Andrew M. Mason, Reg. No. 64,034
andrew.mason@klarquist.com
KLARQUIST SPARKMAN, LLP
One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Tel: 503-595-5300
Fax: 503-595-5301

Counsel for Petitioner

CERTIFICATE OF SERVICE
IN COMPLIANCE WITH 37 C.F.R. § 42.6(e)(4)

The undersigned certifies that the **PETITION FOR *INTER PARTES* REVIEW OF U.S. PATENT NO. 11,256,708, and EXHIBITS 1001 – 1019** were served on October 23, 2026, via **Express Mail** on the Patent Owner at the following address of record as listed on the USPTO Patent Center:

Myers Wolin, LLC
100 South Jefferson Road
Suite 202
Whippany, NJ 07981-1009

By: /Andrew M. Mason/
Andrew M. Mason, Reg. No. 64,034
andrew.mason@klarquist.com
KLARQUIST SPARKMAN, LLP
One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Tel: 503-595-5300
Fax: 503-595-5301

Counsel for Petitioner