# Schritte zu einer zertifizierten Informationsflussanalyse von Geschäftsprozessen

Thomas S. Heinze

Institut für Informatik
Friedrich-Schiller-Universität Jena
t.heinze@uni-jena.de

**Abstract.** This paper introduces an easy to use methodology to analyze the information flow within business processes based on different confidentiality levels. Furthermore, this paper also provides a formal proof based on Coq for this methodology.

**Zusammenfassung.** In diesem Beitrag wird eine einfache Methode zur Analyse des Informationsflusses in Geschäftsprozessen auf Grundlage von Vertraulichkeitsstufen vorgestellt. Die Korrektheit der Analyse wird anhand einer maschinenprüfbaren Formalisierung in Coq nachgewiesen.

## 1 Einführung und Motivation

Die statische Datenflussanalyse ist ein geeignetes Werkzeug zur Unterstützung der Geschäftsprozessmodellierung und -analyse. Wie bereits in [8] ausgeführt, bietet sie nicht nur ein allgemein einsetzbares Rahmenwerk zur Prozessanalyse, sondern beruht gleichzeitig auch auf der wohldefinierten Theorie der *abstrakten Interpretation* [6], die sich leicht zum formalen Korrektheitsnachweis heranziehen lässt. In Verbindung mit modernen maschinengestützten Beweisassistenten gestattet die Formalisierung von Datenflussanalysen auf Grundlage der abstrakten Interpretation schließlich die Definition sogenannter *zertifizierter Analysen*. Grundlage einer solchen zertifizierten Analyse bildet dabei ein maschinenprüfbarer Beweis der Analysekorrektheit, aus dem sich die Analyseimplementierung extrahieren lässt. Der Vorteil liegt dann auf der Hand, da ein Anwender nun nicht mehr auf die Korrektheit der Implementierung vertrauen muss, sondern stattdessen den Korrektheitsbeweis automatisiert nachvollziehen kann. In einem Auditszenario ließe sich derart die Compliance-Prüfung von Geschäftsprozessen [2] durch eine orthogonale Prüfung zur Vertrauenswürdigkeit des Audits selbst ergänzen.

Ein Beispiel für ein solches Szenario ist die Analyse von Informationsflüssen, oder genauer die Untersuchung des Flusses von Informationen in einem Prozessmodell, mit der sich etwa Compliance-Regeln zur vertraulichen Verarbeitung von sensitiven Informationen, beispielsweise Gesundheitsdaten, überprüfen lassen. In diesem Beitrag wird eine erste einfache statische Datenflussanalyse zur Analyse

N. Herzberg, C. Hochreiner, O. Kopp, J. Lenhard (Eds.): 10th ZEUS Workshop, ZEUS 2018, Dresden, Germany, 8-9 February 2018, published at http://ceur-ws.org/Vol-2072

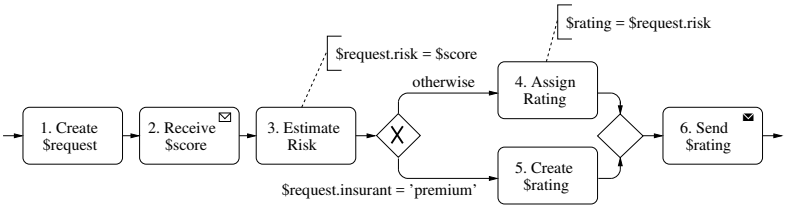


**Abb. 1.** Beispielprozess in BPMN-Notation

des Informationsflusses in Geschäftsprozessen auf Grundlage von Vertraulichkeitsstufen vorgestellt. Die Analyse beruht auf dem Verfahren der Andersen-Analyse [16] und wurde mit Hilfe des Beweisassistenten *Coq*[1] formal definiert, so dass der Nachweis der Analysekorrektheit als maschinenprüfbarer Beweis zur Verfügung steht. Der Beitrag kann somit als ein Schritt zu einer *zertifizierten Informationsflussanalyse* von Geschäftsprozessen verstanden werden.

Im sich anschließenden Abschnitt 2 wird das Analyseszenario genauer abgegrenzt und die Analyse vorgestellt. Die formale Entwicklung und der Korrektheitsnachweis zur Analyse auf Grundlage der Theorie der abstrakten Interpretation erfolgt in Abschnitt 3. In Abschnitt 4 wird ein kurzer Überblick zu verwandten Arbeiten gegeben, bevor der Beitrag mit einer Zusammenfassung und einem Ausblick auf weiterführende Arbeiten in Abschnitt 5 endet.

## 2 Analyse des Informationsflusses

Eine *Informationsflussanalyse* [7] untersucht einen Prozess hinsichtlich des Flusses von Informationen, wobei verschiedene Vertraulichkeitsstufen berücksichtigt werden können. Im einfachsten Fall werden die zwei Stufen $H$ (vertraulich) und $L$ (öffentlich) betrachtet. Zur Sicherstellung der Vertraulichkeit sollte dann kein Informationsfluss von einer mit $H$ ausgezeichneten Quelle an eine mit $L$ ausgezeichnete Senke existieren. In Abbildung 1 ist ein Beispielprozess dargestellt, in dem die Risikobewertung für eine Versicherung bestimmt und versendet wird. Dazu wird zunächst ein Formular in $\$request$ erzeugt und der vertrauliche Score-Wert $\$score$ abgefragt. In Abhängigkeit vom Versichertenstatus wird entweder ein Standardwert oder der Score-Wert als Risikobewertung $\$rating$ verwendet und weitergeleitet. Wird in diesem Beispiel die eingehende Nachricht $\$score$ mit der Vertraulichkeitsstufe $H$ und die ausgehende Nachricht $\$rating$ mit der Stufe $L$ ausgezeichnet, liegt für den oberen Pfad ein Informationsfluss von $H$ nach $L$ und somit eine Verletzung der Informationsvertraulichkeit vor.

Hier soll eine statische Datenflussanalyse zur Untersuchung des Informationsflusses in Prozessmodellen definiert werden. Gegenstand der Analyse sind explizite Datenflüsse, wie im obigen Beispiel, jedoch keine impliziten Informationsflüsse oder Seitenkanäle (vergleiche den impliziten Fluss von $\$request.insurant$

[1] `https://coq.inria.fr`, letzter Zugriff am 8. März 2018

$$
\begin{aligned}
i\colon \mathit{Receive}^{L}\ x/\ i\colon \mathit{Create}^{L}\ x: &\quad o_i^L \in pt(x)\\
i\colon \mathit{Receive}^{H}\ x/\ i\colon \mathit{Create}^{H}\ x: &\quad o_i^H \in pt(x)\\
x = y: &\quad pt(y) \subseteq pt(x)\\
x = y.f: &\quad \frac{o_i^L \in pt(y)}{pt(o_i.f) \subseteq pt(x)} \qquad \frac{o_i^H \in pt(y)}{pt(o_i.f) \subseteq pt(x)}\\
x.f = y: &\quad \frac{o_i^L \in pt(x)}{pt(y) \subseteq pt(o_i.f)} \qquad \frac{o_i^H \in pt(x)}{pt(y) \subseteq pt(o_i.f)}
\end{aligned}
$$

**Abb. 2.** Regelsystem zur statischen Informationsflussanalyse

nach $\$\mathit{rating}$ über die Kontrollabhängigkeit der Variablendefinition). Unter diesen Anforderungen lässt sich die Analyse als *Taint-Analyse* [14] auffassen und auf das Verfahren der Andersen-Analyse [16] zurückführen. Durch dieses Verfahren werden Variablen, und deren Komponenten im Fall zusammengesetzter Variablentypen, Mengen mit abstrakten Objekten zugewiesen. Weiterhin werden Regeln zwischen den Mengen in Form von Element-/Teilmengenbeziehungen definiert. Die Lösung des dadurch charakterisierten Regelsystems ergibt dann einen Fixpunkt als Abschätzung zum Datenfluss im analysisertem Prozess.

In Abbildung 2 sind die Regeln der Analyse für einen kleinen Sprachausschnitt dargestellt. Auf der linken Seite stehen die Prozessaktivitäten und auf der rechten Seite die zugehörigen Regelschemata. Zusätzlich zu einer herkömmlichen Andersen-Analyse wird hier zwischen öffentlichen und vertraulichen Informationsquellen unterschieden, so dass die Aktivitäten *Receive* und *Create* eine entsprechende Auszeichnung mit $L$ oder $H$ aufweisen müssen. Komplexere Sprachkonstrukte, etwa die Zuweisung $x.f.g = y.h$, lassen sich durch die Einführung zusätzlicher Variablen ebenfalls abbilden. Ferner wird von atomaren Nachrichten ausgegangen.

Durch die Regeln wird für jede Prozessaktivität beschrieben, wie deren Ausführung die den Variablen und Komponenten zugeordneten Mengen $pt$ mit abstrakten Objekten beeinflusst. Als Fixpunktlösung ergibt sich für die Regeln dann eine Überabschätzung des Datenflusses, da Ausführungsreihenfolge und -kontext von Aktivitäten unberücksichtigt bleiben (*fluss-/kontextinsensitiv* [16]). Zudem wird pro *Receive*- und *Create*-Aktivität $i$ nur jeweils ein abstraktes Objekt $o_i$ unterschieden, unabhängig davon wie oft die Aktivität ausgeführt wird. Wird der Beispielprozess in Abbildung 1 betrachtet, und von einer vertraulichen Quelle $\$\mathit{score}$ sowie einer öffentlichen Quelle $\$\mathit{request}$ ausgegangen, ergibt sich unter Anwendung der Regeln die folgende Ableitung und somit eine Vertraulichkeitsverletzung, unter Annahme der Vertraulichkeitsstufe $L$ für die ausgehende Nachricht $\$\mathit{rating}$ (Regelanwendungen sind mit der zugehörigen Aktivitäts-Id angegeben):

$$
\dfrac{\dfrac{\dfrac{}{o_1^L \in pt(\$\mathit{request})}\,1}{pt(o_1.\mathit{risk}) \subseteq pt(\$\mathit{rating})}\,4 \qquad \dfrac{\dfrac{\dfrac{}{o_1^L \in pt(\$\mathit{request})}\,1}{pt(\$\mathit{score}) \subseteq pt(o_1.\mathit{risk})}\,3 \qquad \dfrac{}{o_2^H \in pt(\$\mathit{score})}\,2}{o_2^H \in pt(o_1.\mathit{risk})}}{o_2^H \in pt(\$\mathit{rating})}
$$

$$\mathit{Receive}^L\ x/\ \mathit{Create}^L\ x,\ (\mathit{vars}, \mathit{heap}, \mathit{high}, l) \to (\mathit{vars}[x \leftarrow l+1], \mathit{heap}, \mathit{high}, l+1)$$

$$\mathit{Receive}^H\ x/\ \mathit{Create}^H\ x,\ (\mathit{vars}, \mathit{heap}, \mathit{high}, l) \to (\mathit{vars}[x \leftarrow l+1], \mathit{heap}, \mathit{high} \cup \{l+1\}, l+1)$$

$$x = y,\ (\mathit{vars}, \mathit{heap}, \mathit{high}, l) \to (\mathit{vars}[x \leftarrow \mathit{vars}(y)], \mathit{heap}, \mathit{high}, l)$$

$$\frac{\mathit{heap}(\mathit{vars}(y)) \neq 0}{x = y.f,\ (\mathit{vars}, \mathit{heap}, \mathit{high}, l) \to (\mathit{vars}[x \leftarrow \mathit{heap}(\mathit{vars}(y))], \mathit{heap}, \mathit{high}, l)}$$

$$\frac{\mathit{heap}(\mathit{vars}(y)) = 0}{x = y.f,\ (\mathit{vars}, \mathit{heap}, \mathit{high}, l) \to (\mathit{vars}, \mathit{heap}, \mathit{high}, l)}$$

$$\frac{\mathit{vars}(x) \neq 0}{x.f = y,\ (\mathit{vars}, \mathit{heap}, \mathit{high}, l) \to (\mathit{vars}, \mathit{heap}[\mathit{vars}(x) \leftarrow \mathit{vars}(y)], \mathit{high}, l)}$$

$$\frac{\mathit{vars}(x) = 0}{x.f = y,\ (\mathit{vars}, \mathit{heap}, \mathit{high}, l) \to (\mathit{vars}, \mathit{heap}, \mathit{high}, l)}$$

**Abb. 3.** Konkrete Semantik (Prädikat `[exec]` in Coq-Formalisierung)

## 3 Zertifizierte Informationsflussanalyse

Im Folgenden soll die im vorangegangenen Abschnitt vorgestellte Analyse formal definiert und deren Korrektheit bewiesen werden. Die Formalisierung erfolgte mit Hilfe des Beweisassistenten *Coq*, so dass ein maschinenprüfbarer Beweis zur Analysekorrektheit[2] zur Verfügung steht und die Implementierung der Analyse aus dem Beweis extrahiert werden kann. Der Coq-Formalismus beruht dabei wesentlich auf bestehenden Coq-Quellen[3] zur Andersen-Analyse.

Für die formale Entwicklung einer statischen Analyse kann auf die *abstrakten Interpretation* zurückgegriffen werden [6]. In der abstrakten Interpretation wird ein Prozess nicht auf konkreten sondern auf abstrakten Werten ausgeführt. Anstatt mit *int*-Werten wird etwa mit den Werten *odd* und *even* gerechnet, so dass sich die Addition aus den Regeln $\mathit{even} + \mathit{even} = \mathit{even}$, $\mathit{even} + \mathit{odd} = \mathit{odd}$, $\mathit{odd} + \mathit{even} = \mathit{odd}$ und $\mathit{odd} + \mathit{odd} = \mathit{even}$ ergibt. Die Ausführung auf konkreten Werten definiert die *konkrete Semantik* und die Ausführung auf abstrakten Werten die *abstrakte Semantik*. Letztere wird zur Definition der Analyse verwendet, für die ausgehend von einem Startzustand durch kontinuierliche Anwendung der abstrakten Semantik auf einen Prozess ein Fixpunkt errechnet wird. Zum Nachweis der Korrektheit muss gezeigt werden, dass der Fixpunkt die sich für jeden Zustand aus der konkreten Semantik ergebenden konkreten Werte abschätzt.

Zur Formalisierung der Informationsflussanalyse wird die in Abbildung 3 dargestellte *konkrete Semantik* genutzt. Darin werden Variablen, Komponenten und Objekte als natürliche Zahlen kodiert. Ein Zustand $s$ enstpricht einem Tupel $(\mathit{vars}, \mathit{heap}, \mathit{high}, l)$, wobei $\mathit{vars}, \mathit{heap}\colon \mathbb{N} \to \mathbb{N}$ Variablen und Komponenten die Objekte zuordnet, auf die sie im Zustand $s$ verweisen, $\mathit{high} \subseteq \mathbb{N}$ alle vertraulichen Objekte aufzählt und $l \in \mathbb{N}$ das zuletzt erzeugte Objekt bezeichnet. Die konkrete Semantik definiert damit eine Relation $i, s \to s'$ zwischen einem Zustand $s$, einer

[2] `https://gitlab.com/t.heinze/zeus2018.git`

[3] `http://adam.chlipala.net/itp/coq/src`, letzter Zugriff am 8. März 2018

$$
\begin{aligned}
&\mathit{Receive}^L\ x\#k/\ \mathit{Create}^L\ x\#k,\ (\mathit{avars}, \mathit{aheap}, \mathit{ahigh}) \\
&\qquad \to (\mathit{avars}[x \leftarrow \mathit{avars}(x) \cup \{k\}], \mathit{aheap}, \mathit{ahigh}) \\
&\mathit{Receive}^H\ x\#k/\ \mathit{Create}^H\ x\#k\ (\mathit{avars}, \mathit{aheap}, \mathit{ahigh}) \\
&\qquad \to (\mathit{avars}[x \leftarrow \mathit{avars}(x) \cup \{k\}], \mathit{aheap}, \mathit{ahigh} \cup \{k\}) \\
&x = y,\ (\mathit{avars}, \mathit{aheap}, \mathit{ahigh}) \to (\mathit{avars}[x \leftarrow \mathit{avars}(x) \cup \mathit{avars}(y)], \mathit{aheap}, \mathit{ahigh}) \\
&x = y.f,\ (\mathit{avars}, \mathit{aheap}, \mathit{ahigh}) \\
&\qquad \to (\mathit{avars}[x \leftarrow \mathit{avars}(x) \cup \bigcup_{k \in \mathit{avars}(y)} \mathit{aheap}(k)], \mathit{aheap}, \mathit{ahigh}) \\
&x.f = y,\ (\mathit{avars}, \mathit{aheap}, \mathit{ahigh}) \\
&\qquad \to (\mathit{avars}, \mathit{aheap}[k \leftarrow \mathit{aheap}(k) \cup \mathit{avars}(y) \mid k \in \mathit{avars}(x)], \mathit{ahigh})
\end{aligned}
$$

**Abb. 4.** Abstrakte Semantik (Prädikat `[abstract_exec]` in Coq-Formalisierung)

auf diesem ausgeführten Prozessaktivität $i$ und dem sich daraus ergebenden Folgezustand $s'$. Durch die Relation wird beispielsweise für eine Aktivität $\mathit{Create}^H\ x$ im Folgezustand ein neues Objekt $l + 1$ erzeugt, der Menge vertraulicher Objekt hinzugefügt und der Verweis der Variablen $x$ auf das Objekt $l + 1$ gesetzt.

In der *abstrakten Semantik* in Abbildung 4 werden abstrakte Zustände betrachtet, die Tupel $(\mathit{avars}, \mathit{aheap}, \mathit{ahigh})$ sind, wobei $\mathit{avars}, \mathit{aheap}\colon \mathbb{N} \to \mathcal{P}(\mathbb{N})$ nun die möglichen Verweisziele von Variablen und Komponenten in Form von Mengen abschätzen (vergleiche auch Abbildung 2), *ahigh* fasst alle vertraulichen Objekte zusammen. Im Gegensatz zur konkreten Semantik wird nicht mehr für jede Ausführung einer *Receive*- oder *Create*-Aktivität ein neues Objekt erzeugt, sondern für dieselbe Aktivität nur ein Objekt verwendet. Die Aktivitäten werden dazu durchnummeriert, so dass sich das Objekt aus dem Index $k$ ergibt.

Für den Nachweis der Korrektheit werden die konkrete und die abstrakte Semantik in Beziehung gesetzt, dies mittels sogenannter Objektpfade:

$$
\frac{s = (\mathit{vars}, \mathit{heap}, \mathit{high}, l)}{s \vdash v :: \mathit{vars}[v]} \qquad \frac{s = (\mathit{vars}, \mathit{heap}, \mathit{high}, l) \quad s \vdash p :: l' \quad l' \neq 0}{s \vdash p :: l' :: \mathit{heap}[l']}
$$

$$
\frac{a = (\mathit{avars}, \mathit{aheap}, \mathit{ahigh}) \quad k \in \mathit{avars}[v]}{a \vdash v :: k}
$$

$$
\frac{a = (\mathit{avars}, \mathit{aheap}, \mathit{ahigh}) \quad a \vdash p :: k \quad k' \in \mathit{aheap}[k]}{a \vdash p :: k :: k'}
$$

Ein Objektpfad $v :: p :: n$ ist ein Prädikat, das ausgehend von einer Variablen $v$, unter einen gegebenen konkreten Zustand $s$ beziehungsweise abstrakten Zustand $a$, eine eventuell leere Sequenz $p$ von Kompontenenzugriffen auf ein Objekt $n$ beschreibt. Vereinfacht gesprochen bezeichnet ein Objektpfad die Möglichkeit unter Zustand $s$ beziehungsweise $a$ über die Variable $v$ auf das Objekt $n$ zuzugreifen. Wie in Abbildung 5 definiert, ist ein abstrakter Zustand $a$ dann eine Abschätzung für einen konkreten Zustand $s$, falls zusätzlich zu gewissen Nebenbedingungen folgende zwei Bedingungen erfüllt sind: (1) Für jedes Objekt $l'$ auf das im Zustand $s$ über Objektpfade $v_1 :: p :: l'$ und $v_2 :: q :: l'$ zugegriffen werden kann, existiert im Zustand $a$ ein entsprechendes Objekt $k$ auf das über Objektpfade $v_1 :: p' : k$

$a = (avars, aheap, ahigh)$ approximates $s = (vars, heap, high, l)$
$\Leftrightarrow_{df}$ $heap(0) = 0 \wedge \forall l' > l \colon heap(l') = 0 \wedge \forall l' \colon s \vdash p :: l' \Rightarrow l' \leq l \wedge \forall l' \in high \colon l' \leq l$
$\wedge \forall v_1, v_2, l' \neq 0 \colon s \vdash v_1 :: p :: l' \wedge v_2 :: q :: l' \Rightarrow \exists k \colon a \vdash v_1 :: p' :: k \wedge a \vdash v_2 :: q' :: k$
$\wedge \forall v, l' \neq 0 \colon s \vdash v :: p :: l \wedge l \in high \Rightarrow \exists k \colon a \vdash v :: q :: k \wedge k \in ahigh$

Konservativität der abstrakten Semantik:

$\forall a', s', s \colon s' \rightsquigarrow s \wedge a'$ approximates $s' \Rightarrow \exists a \colon a' \rightsquigarrow a \wedge a$ approximates $s$ □

$v$ not $H \Leftrightarrow_{df} \forall s = (vars, heap, high, l) \colon init \rightsquigarrow s \Rightarrow vars(v) \notin high$

Korrektheit der Informationsflussanalyse:

$\forall v \colon (\forall a = (avars, aheap, ahigh) \colon ainit \rightsquigarrow a \Rightarrow \forall k \in avars(v) \colon k \notin ahigh) \Rightarrow v$ not $H$ □

**Abb. 5.** Korrektheit der Analyse (Theorem `[analysis_sound]` in Coq-Formalisierung)

und $v_2 :: q' :: k$ zugegriffen werden kann. (2) Für jedes vertrauliche Objekt $l'$ auf das über einen Objektpfad $v :: p :: l'$ im Zustand $s$ zugegriffen werden kann, existiert im Zustand $a$ ein entsprechendes vertrauliches Objekt $k$ auf das über einen Objektpfad $v :: q :: k$ zugegriffen werden kann. Anhand dieser Definitionen lässt sich zeigen, dass die abstrakte Semantik eine konservative Abschätzung für die konkrete Semantik ist, das heißt für alle erreichbaren konkreten Zustände $s$ existiert ein erreichbarer abstrakter Zustand $a$, der eine Abschätzung für $s$ ist.

Als Folgerung aus dieser Beziehung zwischen konkreter und abstrakter Semantik kann auf die Korrektheit der Informationsflussanalyse geschlossen werden. So lässt sich zeigen, dass für alle Variablen $v$ gilt, falls $v$ unter allen erreichbaren abstrakten Zuständen $a$ auf kein vertrauliches Objekt verweist, so verweist $v$ auch unter allen erreichbaren konkreten Zuständen $s$ auf kein vertrauliches Objekt. Unter Anwendung der Analyse kann somit ein Fluss von einer vertraulichen Quelle an eine öffentliche Senke, in Form einer Variablen $v$, sicher ausgeschlossen werden. Neben der Korrektheit der Analyse kann auch deren Terminierung gezeigt werden. Aus Platzgründen wird auf eine entsprechende Diskussion verzichtet und stattdessen auf die Coq-Formalisierung (Definition `[fixed_point]`) verwiesen.

## 4 Verwandte Arbeiten

In [8] wurde der Ansatz zur zertifizierten Analyse von Geschäftsprozessen zunächst allgemein motiviert, in diesem Beitrag konnte die Machbarkeit des Ansatzes nun konkret am Beispiel der Analyse des Informationsflusses von Geschäftsprozessen gezeigt werden. Die Prüfung des Informationsflusses ist dabei ein Standardproblem der statischen Analyse [7]. Es gibt zahlreiche Varianten, die neben expliziten Datenflüssen auch Seitenkanäle, zum Beispiel das Zeitverhalten oder den Energieverbrauch, einbeziehen. Informationsflussanalysen werden neben dem hier betrachteten Aufdecken von Informationslecks insbesondere zur Identifizierung von Sicherheitslücken eingesetzt, oft in Form einer *Taint-Analyse* [14]. Eine geläufige Technik zur Taint-Analyse ist das Andersen-Verfahren [16]. Für dieses wird zwischen inter- und intraprozeduralen, fluss- und kontextsensitiven sowie

-insensitiven Varianten unterschieden. Das hier beschriebene Verfahren setzt eine sehr einfache intraprozedurale und fluss-/kontextinsensitive Analyse um.

Der Einsatz von Beweisassistenten zur Verifikation von statischen Analysen erfährt eine stetig wachsende Verbreitung. Insbesondere *Coq* spielt eine herausragende Rolle, wie nicht zuletzt an der erfolgreichen Verifikation eines realistischen optimierenden Compilers im *CompCert*-Projekt [13] deutlich wird. Im Allgemeinen wird zum Nachweis der Analysekorrektheit analog dem Vorgehen in diesem Beitrag auf die Theorie der abstrakten Interpretation zurückgegriffen [4,5]. Neben den auch dieser Arbeit zugrundeliegenden Coq-Quellen zum Andersen-Verfahren von Adam Chlipala existieren weitere entsprechende Formalisierungen [11,15].

Eine Reihe von typischerweise petrinetzbasierten Techniken zur Analyse des Informationsflusses wird auch für Geschäftsprozesse beschrieben, eine Übersicht kann etwa [1] entnommen werden. Die hier vorgestellte Analyse bezieht sich dabei auf den expliziten Datenfluss und Vertraulichtkeitsstufen (*Mandatory Access Control*). Vergleichbare Arbeiten für Geschäftsprozesse beruhen meist auf höheren Petrinetzen und Methoden des Model-Checking [3,12]. Damit ergeben sich aber zwei Problemstellungen, die für den in dieser Arbeit beschriebenen Ansatz nicht auftreten. So stellen sich für das Model-Checking auf höheren Petrinetzen aufgrund potentieller Zustandsraumexplosion grundlegende Skalierungsprobleme. Durch eine geschickte Wahl der Abstraktion in den Petrinetzmodellen lassen sich diese zwar prinzipiell umgehen, jedoch bedingt dies dann eine aufwendige und nicht triviale manuelle Modellierung der zu analysierenden Geschäftsprozesse. Ferner ist dem Autor kein Ansatz zur zertifizierten Informationsflussanalyse, das heißt maschinell verifizierten Analyse, für Geschäftsprozesse bekannt.

## 5 Zusammenfassung und Ausblick

In diesem Beitrag wird eine einfache statische Analyse zur Untersuchung des Informationsflusses in Geschäftsprozessen vorgestellt. Mit Hilfe der Analyse lassen sich Datenflüsse von sensitiven Datenquellen an öffentliche Senken für einen Prozess sicher ausschließen. Die Analyse beruht auf dem Verfahren der Andersen-Analyse und ist mittels der Theorie der abstrakten Interpretation formalisiert, so dass sich Korrektheit und Terminierung im Beweisassistent *Coq* maschinenprüfbar beweisen lassen. Da zudem die Möglichkeit besteht, die Analyseimplementierung aus dem Beweis zu extrahieren, handelt es sich um einen ersten Schritt zu einer *zertifizierten Informationsflussanalyse* für Geschäftsprozesse.

In weiterführenden Arbeiten sollen Fragen zur Präzision und Skalierbarkeit der Analyse untersucht werden. So kann die Präzision durch Definition einer flusssensitiven Analyse erhöht werden, etwa unter Verwendung *erweiterter Workflow-Graphen* [9,10]. In diesem Fall ist aber eine Formalisierung der Transformation in erweiterte Workflow-Graphen in *Coq* notwendig.

**Danksagung.** Ich danke Adam Chlipala für die Bereitstellung der Coq-Quellen zur Andersen-Analyse.

## Literatur


1. Accorsi, Rafael ; Lehmann, Andreas ; Lohmann, Niels: Information leak detection in business process models: Theory, application, and tool support. In: *Information Systems* 47 (2015), S. 244–257
2. Accorsi, Rafael ; Lowis, Lutz ; Sato, Yoshinori: Automated Certification for Compliant Cloud-based Business Processes. In: *Business & Information Systems Engineering* 3 (2011), Nr. 3, S. 145–154
3. Barkaoui, Kamel ; Ayed, Rahma B. ; Boucheneb, Hanifa ; Hicheur, Awatef: Verification of Workflow Processes Under Multilevel Security Considerations. In: *Proceedings, Third International Conference on Risks and Security of Internet and Systems, CRiSIS 2008, Tozeur, Tunisia, October 28-30, 2008*, IEEE, 2008, S. 77–84
4. Bertot, Yves: Structural Abstract Interpretation: A Formal Study Using Coq. In: *Language Engineering and Rigorous Software Development, International LerNet ALFA SummerSchool 2008, Piriapolis, Urugay, February 24-March 1, 2008, Revised Tutorial Lectures*. Springer, 2009 (LNCS 5520), S. 153–194
5. Besson, Frédéric ; Cachera, David ; Jensen, Thomas ; Pichardie, David: Certified Static Analysis by Abstract Interpretation. In: *Foundations of Security Analysis and Design V*. Springer, 2009 (LNCS 5705), S. 223–257
6. Cousot, Patrick ; Cousot, Radhia: Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs. In: *Proceedings of the 4th ACM Symposium on Principles of Programming Languages*, ACM, 1977, S. 238–252
7. Denning, Dorothy E.: A Lattice Model of Secure Information Flow. In: *Communications of the ACM* 19 (1976), Nr. 5, S. 236–243
8. Heinze, Thomas S.: Towards Certified Data Flow Analysis of Business Processes. In: *9th ZEUS Workshop, ZEUS 2017, Lugano, Switzerland, 13–14 February 2017, Proceedings*, CEUR-WS.org, 2017 (CEUR Workshop Proceedings 1826), S. 1–3
9. Heinze, Thomas S. ; Amme, Wolfram ; Moser, Simon: A Restructuring Method for WS-BPEL Business Processes Based on Extended Workflow Graphs. In: *Business Process Management, 7th International Conference, BPM 2009, Ulm, Germany, September 8-10, 2009, Proceedings*, Springer, 2009 (LNCS 5701), S. 211–228
10. Heinze, Thomas S. ; Amme, Wolfram ; Moser, Simon: Static analysis and process model transformation for an advanced business process to Petri net mapping. In: *Software: Practice and Experience* 48 (2018), Nr. 1, S. 161–195
11. Jourdan, Jacques-Henri ; Laporte, Vincent ; Blazy, Sandrine ; Leroy, Xavier ; Pichardie, David: A Formally-Verified C Static Analyzer. In: *ACM SIGPLAN Notices* 50 (2015), Nr. 1, S. 247–259
12. Juszczyszyn, Krzysztof: Verifying Enterprise's Mandatory Access Control Policies with Coloured Petri Nets. In: *12th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, IEEE, 2003, S. 184–189
13. Leroy, Xavier: Formal Verification of a Realistic Compiler. In: *Communications of the ACM* 52 (2009), Nr. 7, S. 107–115
14. Livshits, V. B. ; Lam, Monica S.: Finding Security Vulnerabilities in Java Applications with Static Analysis. In: *Proceedings of the 14th USENIX Security Symposium*, USENIX Association, 2005, S. 271–286
15. Robert, Valentin ; Leroy, Xavier: A Formally-Verified Alias Analysis. In: *Certified Programs and Proofs, Second International Conference, CPP 2012, Kyoto, Japan, December 13-15, 2012, Proceedings*, Springer, 2009 (LNCS 7679), S. 11–26
16. Sridharan, Manu ; Chandra, Satish ; Dolby, Julian ; Fink, Stephen J. ; Yahav, Eran: Alias Analysis for Object-Oriented Programs. In: *Aliasing in Object-Oriented Programming*. Springer, 2013 (LNCS 7850), S. 196–232