JOCHEN RICHNER
ALEXANDER BERNAUER

REGULATION AS CODE

Effektive Kommunikation zwischen Steuer- und Technologiefachpersonen in der Praxis?

Die Steuerwelt wird zunehmend digitalisiert: Kommunikation und Datenverarbeitung finden bei vielen Unternehmen weitgehend digital statt mit einer wachsenden Automatisierung. Umso mehr geraten die Bereiche in den Fokus, in denen weiterhin Fachpersonen manuell arbeiten müssen. Die Verständigung und die Auslegung von Gesetzestexten – besonders von hochkomplexen wie BEPS 2.0 – bietet sich da an. Die Verlockung ist gross, die Herausforderungen sind es auch.

1. EINLEITUNG

Seit Jahren werden Kommunikation und Datenverarbeitung auch im Steuerbereich zunehmend digitalisiert und automatisiert. Damit einher geht idealerweise eine Steigerung von Geschwindigkeit und Qualität. Die rechtlichen Grundlagen sind aber bisher von diesem Trend «verschont» geblieben. Sie werden weiterhin in menschlicher Sprache verfasst, was Zweideutigkeiten, Widersprüchlichkeiten und Unvollständigkeiten mit sich bringt.

Hier setzt Regulation as Code an: Das Ziel ist, gesetzliche Regelungen nicht nur in menschlich lesbarer, sondern gleichzeitig in maschinenverarbeitbarer Sprache (Programmcode) auszudrücken [1]. Der vorliegende Artikel zeigt die Vor- und Nachteile von Regulation as Code auf und gibt einen Einblick in konkrete Anwendungsfälle.

2. HINTERGRUND

2.1 Ausgangslage. Sprache in der Rechtssetzung untersteht einem letztlich nicht zu lösenden Zielkonflikt: Einerseits soll sie möglichst präzis sein, andererseits muss sie alle denkbaren Anwendungsfälle abdecken – und gleichzeitig soll sie möglichst einfach verständlich sein.

Die Ausgangslage wird verschärft durch eine wachsende Komplexität und Beschleunigung des Gesetzgebungsprozesses. Globale Projekte wie BEPS 2.0 illustrieren dies. Es resultieren häufig Rechtsunsicherheit sowie hohe Kosten bei Steuerzahlenden und Behörden bei der Anwendung.

2.2 Anwendungsbereiche. Die akademische Diskussion wird dominiert von der staatlichen Perspektive: Gesetzgeber sollen Rechtstexte (auch) in Programmcode erlassen, um eine maschinelle Verarbeitung durch Rechtssubjekte und Exekutivbehörden zu ermöglichen. Die Vorteile dafür liegen auf der Hand: erhöhte Präzision durch den Wegfall von jeder der menschlichen Sprache inhärenten Ungenauigkeit und Effizienzsteigerungen durch maschinelle Verarbeitung. Daneben werden auch demokratische und rechtsstaatliche Elemente als Vorteile angeführt: Rechts- sowie Steuerfachpersonen sollen als «Übersetzende» zwischen Gesetzgeber und Bürgerinnen bzw. Bürgern überflüssig werden [2]. Als führend wird hierbei Neuseeland angesehen, wo bereits erste Anwendungsfälle im Bereich der sozialen Wohlfahrt publiziert wurden [3].

Es kann an dieser Stelle offenbleiben, wie realistisch der Erlass von Regelwerken wie BEPS 2.0 in maschinenverarbeitbarer Form in absehbarer Zukunft in der Schweiz ist. Stattdessen liegt der Fokus dieses Artikels auf den Möglichkeiten, die Regulation as Code dem privaten Sektor im Rahmen eigener «Übersetzungsarbeit» bietet.

3. AKTUELLE ANWENDUNGSBEISPIELE

3.1 Vorteile. Zunächst sind die zentralen Vorteile von Regulation as Code festzuhalten:

→ Zuverlässigkeit: Dank maschinenverarbeitbarem Format können in Regulation as Code verfasste Regeln auch bei mehrfacher Ausführung auf vergleichbare Sachverhalte gleich an-

JOCHEN RICHNER,
LIC. IUR.,
DIPL. STEUEREXPERTE,
DIRECTOR, LEITER
TAX TECHNOLOGY
PWC SCHWEIZ

ALEXANDER BERNAUER,
DR. SC. ETH ZÜRICH,
UNABHÄNGIGER BERATER,
TECHNOLOGIEEXPERTE,
SOFTWAREINGENIEUR,
FOKUS INTERDISZIPLINÄRE
ANWENDUNGEN

gewendet werden. Unterschiedliche Interpretationen durch unterschiedlich Nutzende – insbesondere mit unterschiedlichen Muttersprachen – entfallen.
→ Testbarkeit: Da die Regeln als Programmcode vorliegen, können sie mit automatisierten Tests und Beispielszenarien auf Korrektheit und Vollständigkeit geprüft werden.
→ Effizienz: Statt bei jeder Rechtsanwendung erneut die (aktuelle) Auslegung prüfen zu müssen, erlaubt Regulation as Code eine schnelle Anwendung von Regeln.
→ Skalierbarkeit: Einmal erfasst, können die Regeln auf eine beliebige Anzahl von Anwendungsfällen übertragen werden.
→ Transparenz: Entscheidungen und Interpretationen, die bei der Übersetzung von Regeln in maschinenverarbeitbares Format erfolgen, sind transparent und können von unabhängiger Stelle überprüft werden. Das gilt insbesondere für Prüfende, welche die einer Berechnung zugrunde liegenden Regeln konkret nachvollziehen können und Gewissheit haben, dass es keine versteckten Annahmen gibt.
→ Plattformunabhängigkeit: Auch wenn dies konzeptionell nicht zwingend ist, hat Regulation as Code zum Ziel, die codierten Regeln innerhalb unterschiedlicher Systeme einsetzen zu können.
→ Versionierung: Regulation as Code ermöglicht das Historisieren von Regeln. Sprich: Es bleibt transparent, welche Regulierungsversion auf welchen Fall angewendet wurde.

3.2 Nachteile. Demgegenüber weist Regulation as Code auch Nachteile auf:
→ Übersetzungsaufwand: d. h. der initiale Aufwand zur Übersetzung der Regeln in maschinenverarbeitbare Form. Dieser fällt prinzipiell auch an, wenn die Regeln in einem Excel Sheet abgebildet werden. Nur wird solche Arbeit typischerweise bedarfsgetrieben und daher ausschnittsweise gemacht, während Regulation as Code einen holistischen Anspruch hat.
→ Risiko: Die Kehrseite der Skalierbarkeit besteht darin, dass Fehler im Code unbemerkt auf eine grosse Anzahl Anwendungsfälle übertragen werden.
→ Personal: Regulation as Code setzt den Einsatz von Mitarbeitenden aus unterschiedlichen Disziplinen voraus, welche eine gewisse Affinität zum jeweils anderen Bereich aufweisen müssen. Erfahrungsgemäss sind solche Spezialistinnen und Spezialisten nicht immer einfach zu finden.

3.3 BEPS 2.0 als möglicher Anwendungsfall. Regulation as Code eignet sich besonders für komplexe Regelwerke, die wiederkehrend auf viele Sachverhalte angewendet werden müssen. Hier bietet sich aktuell Pillar 2 von BEPS 2.0 [4] an: Das komplexe Regelwerk betrifft weltweit Hunderte von Unternehmen, welche zunächst die Auswirkungen auf sie verstehen und danach unternehmensinterne und -externe Vorgänge entsprechend planen sowie regelmässig entsprechende Deklarationen einreichen müssen.

Neben dem Investitionsbedarf aufgrund des initialen Übersetzungsaufwands bestehen primär zwei Vorbehalte gegen Regulation as Code in diesem Bereich:
→ Dynamik: Bereits heute ist absehbar, dass BEPS 2.0 über die nächsten Jahre einer hohen Dynamik unterliegen wird. Nationale Gesetzgeber, die Rechtsprechung sowie Praktikerinnen und Praktiker werden über die Zeit Lücken füllen und Auslegungen weiterentwickeln. Entsprechend ist bis heute bei konkreten Implementierungsprojekten teils eine gewisse Zurückhaltung zu spüren. Hier kann Regulation as Code einen wertvollen und kosteneffizienten Beitrag leisten: Wie in der Softwareentwicklung üblich, können in Regulation as Code abgefasste Regeln unterschiedliche Versionen aufweisen. Bestehende Versionen müssen lediglich punktuell überarbeitet und sich daraus ergebende Änderungen können transparent dargestellt werden.

→ Nationale Besonderheiten: Mit den Model Rules der OECD besteht prinzipiell eine einheitliche Basis. Diese muss jedoch von den einzelnen Ländern in nationale Gesetze übersetzt werden – vergleichbar mit dem Erlass kantonaler Steuergesetze basierend auf dem Steuerharmonisierungsgesetz des Bundes. Dabei können bewusste oder unbewusste Abweichungen zum Standard entstehen. Diese Uneinheitlichkeit, z. B. unterschiedliche Bedeutungen des gleichen Begriffs, stellt eine besondere Herausforderung für technologiegestützte Lösungen dar. Regulation as Code kann helfen, Gemeinsamkeiten und Unterschiede in der Codebasis transparent zu machen.

4. EIN KONKRETES BEISPIEL

4.1 Basis. Zur Übersetzung von Regelwerken in ein maschinenlesbares Format eignet sich z. B. die Programmiersprache Eiger [5]. Dabei handelt es sich um eine domänenspezifische Programmiersprache basierend auf Haskell [6]. Erste Erfahrungen anhand von Pillar 2 zeigen, dass Eiger einerseits eine effiziente Übersetzung von in menschlicher Sprache erfassten Regeln in Code durch Steuerspezialistinnen und Steuerspezialisten erlaubt. Dies insbesondere, weil der Code für Benutzende ähnlich funktioniert wie Excel-Formeln mit einfachen, SQL-ähnlichen-Abfragen [7]. Andererseits ermöglicht Eiger einen plattformunabhängigen Einsatz.

4.2 Regeln schreiben. Die Methodik, die Eiger zugrunde liegt, sieht vor, dass eine Steuerfachperson und eine Software-Ingenieurin bzw. ein Software-Ingenieur gemeinsam Regeln implementieren. Durch die Ingenieurin bzw. den Ingenieur gestellte Fragen zu den Regeln werden durch die Steuerfachperson beantwortet. Die Ingenieurin bzw. der Ingenieur schreibt die Fragen auf, und die Fachperson liest sie. Bei Fehlern oder nötigen Anpassungen startet das Duo den Zyklus von Neuem. Die Fachperson muss den Code nur lesen (und verstehen) und davon auch nur die für die Regeln relevanten Teile, während sich die Ingenieurin bzw. der Ingenieur um das Schreiben und die technische Integration ins Gesamtsystem kümmert.

Aufgrund der mit Eiger gemachten Erfahrung gehen die Autoren davon aus, dass die Formulierung als Code zusammen mit den Fähigkeiten der Ingenieurin bzw. des Ingenieurs, Spezialfälle systematisch aufzudecken, dazu führen, dass die Fachperson sich ein präziseres und umfangreicheres Wissen über das Regelwerk aneignet. Angereichert mit Beispielszenarien, die automatisch getestet werden, entsteht ein zuverlässiges und flexibel einsetzbares Regulation-as-Code-Modul.

Aber wie sehen in Eiger erfasste Regeln konkret aus? Regeln beziehen sich auf Fakten. Solche können z. B. in Zahlen, booleschen Bedingungen, Prozentsätzen o. Ä., aber auch in domänenspezifischen Klassifikationen wie dem Typ einer juristischen Person bestehen. Jedes Faktum kann als Eingabewert manuell oder über eine Schnittstelle in einem Vorsystem bereitgestellt werden. Alternativ existieren Regeln, welche das Herleiten des fraglichen Faktums aus anderen Fakten festlegen. Auf diese Weise bilden Fakten und Regeln einen gerichteten, azyklischen Graphen.

4.3 Regeln ausführen. Sobald das Regulation-as-Code-System mit Regeln und Eingabefakten versorgt wurde, können Steuerfachpersonen das System nach den Werten beliebiger Fakten abfragen. Liegt der Wert eines Faktums bereits vor, wird dieser angezeigt. Gibt es eine Regel, so wird diese ausgeführt. Andernfalls liegt ein Fehler vor, den das System meldet.

Dass existierende Werte den Vorrang vor der Ausführung von Regeln haben, beruht auf der praktischen Relevanz für Simulationen: So können ein Wert für ein entsprechendes Faktum angegeben sowie die Regel und die davon abhängigen Fakten übergangen werden. Bei BEPS könnte man z. B. simulieren, wie sich die Gesamtsteuerbelastung der Gruppe entwickeln würde, wenn eine bestimmte Gesellschaft einen gewissen Gewinn erzielen würde.

Ein weiteres Anschauungsbeispiel für die Flexibilität des Regulation-as-Code-Systems ist das Thema der Elections, also der Wahlmöglichkeiten. So kann eine Gruppe z. B. die finanziell optimale Kombination von Wahlmöglichkeiten ermitteln. Die notwendigen Eingabefakten vorausgesetzt, lässt sich das System mit einem Baustein ergänzen, welcher alle möglichen Wahlmöglichkeiten durchspielt und das System jeweils nach der resultierenden Steuerbelastung fragt.

5. FAZIT

Regulation as Code steckt aktuell noch in den Kinderschuhen. Besonders komplexe, internationale Initiativen wie BEPS 2.0, aber auch mannigfaltige ESG-Regulatorien lassen das Potenzial erkennen, welches dieser Ansatz birgt. Zentral ist, dass Steuerspezialistinnen und Steuerspezialisten über den eigenen Tellerrand schauen und sich auf technische Diskussionen mit Ingenieurinnen und Ingenieuren einlassen. ■

Fussnoten: 1) De Sousa Tim, Abschnitt 1. **2)** Mohun James / Roberts Alex, S. 12. **3)** Mohun James / Roberts Alex, S. 22. **4)** Das OECD-Projekt zur Verhinderung der Gewinnverkürzung und Gewinnverlagerung (BEPS, Base Erosion and Profit Shifting) besteht vereinfacht gesagt aus zwei Bereichen: Pillar 1 betrifft die Neuzuweisung von Besteuerungsrechten. Mit Pillar 2 soll eine globale Mindestbesteuerung von 15 % für grosse, multinationale Unternehmensgruppen eingeführt werden, welche auf einer einheitlichen Steuerbasis beruhen würde. **5)** Bernauer Alexander / Eisenberg Richard. **6)** https://www.haskell.org/. **7)** SQL ist eine Datenbanksprache zur Definition von Datenbankstrukturen und erlaubt das Bearbeiten und Abfragen von so gespeicherten Datensätzen.

Literatur: ▶ Mohun James / Roberts Alex: OECD Working Papers on Public Governance No. 42 – Cracking the code: Rulemaking for humans and machines, https://oecd-opsi.org/wp-content/uploads/2022/03/rac-wp.pdf, zuletzt besucht am 2. September 2022. ▶ De Sousa Tim: Rules as Code Handbook, https://github.com/Rules-as-Code-League/RaC-Handbook/wiki/1-Introduction:-What-is-Rules-as-Code%3F, zuletzt besucht am 2. September 2022. ▶ Bernauer Alexander / Eisenberg Richard A.: Eiger: Auditable, executable, flexible legal regulations, Haskell Symposium 2022 (Systemdemonstration), https://arxiv.org/abs/2209.04939; https://www.youtube.com/watch?v=HDw-eLfPopg.