taxnotes® international

Volume 114, Number 13 ■ June 24, 2024

The Microsoft Case: Lessons for Post-BEPS Software Development Cost Contribution Arrangements

by Oliver Treidler, Tom-Eric Kunz, Michael Dorner, and Maximilian Capraro

Reprinted from *Tax Notes International*, June 24, 2024, p. 1883

The Microsoft Case: Lessons for Post-BEPS Software Development Cost Contribution Arrangements

by Oliver Treidler, Tom-Eric Kunz, Michael Dorner, and Maximilian Capraro

Oliver Treidler

Tom-Eric Kunz

Michael Dorner

Maximilian Capraro

Oliver Treidler is an economist specializing in transfer pricing, Tom-Eric Kunz is a researcher with the Europa-Universität Viadrina Institute for Central and East European Taxation, and Michael Dorner and Maximilian Capraro are software engineers and researchers. They are cofounders of Kolabri GmbH, which is based in Germany and dedicated to consulting on accelerating collaborative software engineering and its compliance with the arm's-length standard.

In this article, the authors examine how the particulars of collaborative software development affect how to structure a software development cost contribution arrangement so that it complies with arm's-length considerations.

Cost contribution arrangements (CCAs) feature among the most complex transfer pricing transactions, presenting myriad challenges for tax and transfer pricing practitioners. The 2022 OECD transfer pricing guidelines[1] devote all of Chapter VIII to this subject and feature detailed examples and explanations to ensure CCAs are commensurate with the arm's-length principle. It is crucial to measure contributions to a CCA by individual entities and determine whether balancing payments are needed to align the contributions with the proportional benefits derived from the CCA by each participant. In accurately delineating the transaction for transfer pricing purposes, it is crucial to have a sound understanding of the business model pursued by the entities participating in a CCA, as well as of the relations and dependencies existing between them.

Failure to appropriately align contributions and benefits can trigger severe tax risks and transfer pricing adjustments. On October 11, 2023, Microsoft Corp. published an update about its

[1]OECD, "Transfer Pricing Guidelines for Multinational Enterprises and Tax Administrations" (2022).

ongoing audit with the Internal Revenue Service; the IRS alleges that Microsoft owes an additional $28.9 billion in tax for 2004 to 2013, plus penalties and interest relating to cost sharing for developing intellectual property between Microsoft's U.S.-based headquarters and foreign subsidiaries.[2] Although detailed information on the audit is not publicly available, Stephen L. Curtis and Reuven S. Avi-Yonah published a detailed account of the Microsoft case in *Tax Notes International* in March 2023.[3] Most of their analysis touches upon the interpretation of IRS-specific regulations on cost sharing[4] dating back to 2008 and earlier. From an international transfer pricing perspective, many aspects of their article may be difficult to interpret. Their careful review and interpretation of the economic facts and circumstances, however, offer a unique opportunity to interpret the compliance of the Microsoft CCA from the perspective of the transfer pricing guidelines.[5] Another unique feature of the Microsoft CCA is that it relates to collaborative software development. Considering that many companies have adopted increasingly decentralized approaches to software development, CCAs for software IP can reasonably be assumed to feature prominently in future tax and transfer pricing audits. Our article provides a contemporary, general understanding of the technical properties inherent in collaborative software development and offers insights into how the economic value contributions by individual entities can be gained, which in turn will inform decisions on how to structure a CCA that complies with arm's-length considerations.

Although we do not offer an assessment regarding the viability of the arguments presented by Curtis and Avi-Yonah from a tax perspective, we will show that some of the working assumptions underlying their arguments may not be aligned with the contemporary characteristics of collaborative software engineering and should be further stress tested from a transfer pricing perspective. We also provide comments and a technical background on how to attain more fact-based, quantitative insights into contributions of individual entities to the development of software-related IP. We provide a data-driven approach to substantiate collaboration and technical dependencies between software subsystems within a multinational enterprise.

The Basics — The OECD on CCAs

Although CCAs can be technically complex, it is sufficient to address three core tenets to follow the interpretations and arguments outlined in this article.

First, the basic rationale of a CCA is defined by paragraph 8.3 of the OECD transfer pricing guidelines: "A CCA is a contractual arrangement among business enterprises to share the contributions and risks involved in the joint development, production or the obtaining of intangibles, tangible assets or services with the understanding that such intangibles, tangible assets or services are expected to create benefits for the individual businesses of each of the participants."

Second, according to paragraph 8.5 of the guidelines, "A key feature of a CCA is the sharing of contributions. In accordance with the arm's length principle, at the time of entering into a CCA, each participant's proportionate share of the overall contributions to a CCA must be consistent with its proportionate share of the overall expected benefits to be received under the arrangement."

The guidelines differentiate between two kinds of CCAs, namely a service CCA and a development CCA. For this article, we focus on the latter, which is generally the more complex of

[2] *See* Daniel Goff, "An Update on Our IRS Tax Audit," Microsoft Blog (Oct. 11, 2023). *See also* Alexander F. Peter, "Microsoft on the Hook for $29 Billion Tax, SEC Filing Shows," *Tax Notes Int'l*, Oct. 16, 2023, p. 439; Ryan Finley, "IRS Hopes to Slay Microsoft's CSA Monster, but How?" *Tax Notes Int'l*, Oct. 23, 2023, p. 479.

[3] Curtis and Avi-Yonah, "Microsoft's Cost-Sharing Arrangement: Frankenstein Strikes Again," *Tax Notes Int'l*, Mar. 6, 2023, p. 1237.

[4] The IRS regulations, and thus Curtis and Avi-Yonah, refer to the term cost-sharing arrangement (CSA), but for the purpose of our article, we apply the terminology of the transfer pricing guidelines: cost contribution arrangements (CCAs). For the questions addressed in the context of this article, we assume that CSA and CCA can be used synonymously.

[5] Curtis and Avi-Yonah, *supra* note 3, and others find that "Microsoft's CSA did not comply with the transition rule in reg. section 1.482-7(m)(1) of the 2008 CSA regulations," referencing reg. section 1.482-7A of the former regulations published in 1996. Our contributions solely relate to an interpretation in the context of the transfer pricing guidelines; any assessment or interpretation of IRS provisions is out of scope of this article.

the two. According to paragraph 8.11 of the guidelines:

> Under a development CCA, each participant has an entitlement to rights in the developed intangible(s) or tangible asset(s). In relation to intangibles, such rights often take the form of separate rights to exploit the intangible in a specific geographic location or for a particular application. . . . In cases where a participant has such rights in any property developed by the CCA, there is no need for a royalty payment or other further consideration for the use of the developed property consistent with the interest to which the participant is entitled under the CCA (however, the contributions of a participant may need to be adjusted if they are not proportionate to their expected benefits).

Third, for a legal entity to qualify as a participant in a CCA, certain conditions must be met. Because of the fundamental importance of the concept of mutual benefit, only an entity that reasonably expects to benefit from the CCA can qualify as a participant. Also, to qualify, such an entity must have control over the risks it assumes under the CCA and have the financial capacity to assume these risks (paragraphs 8.14 and 8.15 of the transfer pricing guidelines). The contributions of the participants of a CCA will in most cases differ in terms of their nature, scope, and quality. Considering that the CCA is designed to pool resources of different parties in an effort to realize synergies, the existence of such differences should be regarded as a characteristic feature of a CCA. From a transfer pricing and tax perspective, however, it needs to be understood that the different contributions must be valued. Without such a valuation it is impossible to align the contributions with the proportionate share of the benefit for each participant and thus ascertain arm's-length conditions.

For development CCAs, specifically those focused on software, the typical contributions consist of performance of ongoing development activities, coordinative activities, and preexisting intangibles. For transfer pricing purposes it is crucial to attain an appropriate and robust valuation. As outlined by the OECD, in cases in which the value of a participant's share of overall contributions is not consistent with that participant's share of expected benefits, the contributions made by at least one of the participants will be inadequate, and the contributions made by at least one other participant will be excessive (paragraph 8.34 of the transfer pricing guidelines). Such an imbalance would be inconsistent with the concept of mutual benefit and thus with the arm's-length principle. Therefore, a CCA will often feature so-called balancing payments by participants to top up the value of their contributions (paragraph 8.35 of the transfer pricing guidelines) and thus ensure proportionality to expected benefits.

In a nutshell, to ascertain compliance with the arm's-length principle, MNEs implementing a CCA need to focus on operationalizing the concept of mutual benefit. An in-depth understanding of how collaborative software development is organized and how synergies are created within the MNE is the foundation of any respective analysis. And, as we show below, only access to robust data reflecting the value creation will allow for a reliable assessment and substantiation of whether the contributions and benefits of CCA participants are proportionate.

Microsoft — CCA for Software Development

This section provides a concise description of the economic background of the Microsoft CCA.[6]

Economic Substance of Foreign CCA Participants

The vantage point adopted by Curtis and Avi-Yonah on the Microsoft case is dominated by scrutinizing abusive tax structuring. Although evaluating potentially abusive tax structuring is outside our scope, one must look at the economic substance of the CCA participants. Doing so clarifies that the CCA participants are not

[6]For the purpose of this article, we narrowly focus on issues relevant to discussing the general transfer pricing implications for CCAs. Notably, we do not discuss any predecessor companies or other legal structuring issues that are of secondary importance outside of the legal questions addressed by Curtis and Avi-Yonah, *supra* note 3. Like Curtis and Avi-Yonah, we did not have access to detailed information on Microsoft's transfer pricing setup. Consequently, no claim or inference is made that our article accurately describes the Microsoft transfer pricing structure. For illustrative purposes, the level of detail of the structure provided by Curtis and Avi-Yonah seems sufficient.

stereotypical letterbox companies. Understanding the scale and scope of the local operations, however, provides necessary orientation for discussing the economic relevance of contributions made by the participants in the CCA.

According to Curtis and Avi-Yonah, Microsoft Ireland (MIR) joined the global CSA in 1999 and made a $7 billion buy-in payment.[7] According to Curtis and Avi-Yonah's sources, MIR had about 2,000 employees of 71 different nationalities. The scale of local operations of MIR, including software development, testing, and localization, is thus reasonably assumed to be complex. The local administrative support functions, such as finance, human resources, and sales and marketing for Europe, the Middle East, and Africa also appear consistent with the economic substance.[8] Curtis and Avi-Yonah emphasized that MIR conducts no internal production operations and instead outsources production to unrelated parties. However, from a general transfer pricing perspective, referring to the basics on CCAs outlined in the section above, outsourcing production would not disqualify MIR as a participant of a CCA. As stipulated in paragraph 8.17 of the OECD transfer pricing guidelines, "it is not necessary for the CCA participants to perform all of the CCA activities through their own personnel. In some cases, the participants in a CCA may decide to outsource certain functions." It would be important, however, to validate whether MIR is assuming all relevant economic functions and risks related to the outsourcing, including the control over the risk pertaining to the outsourced activities.[9] If MIR acts as the entrepreneur for those outsourced activities, it would generally appear consistent that any relevant output from these outsourced activities are considered contributions of MIR to the CCA.

Regarding Singapore-based Microsoft Asia Island Ltd. (MAIL), Curtis and Avi-Yonah explain that it joined the CSA in 2004 by making a buy-in payment of $4 billion and that MAIL had "several hundred" employees (as of 2012) and owned and operated several data centers.[10] It thus seems uncontested that MAIL had sufficient economic substance to distribute Microsoft's software to customers in its territory (Asia) as well as to conduct localized regional production, marketing, and administrative functions. Looking at the setup of MAIL, the existence of several hundred employees and several data centers arguably does not accurately reflect the economic substance on the ground. The MAIL homepage of states that it has a team of some 30,000 people — across sales, marketing, operations, engineering, and development, including the two largest research centers in China and India. According to Microsoft, it operates 20 data centers in the region (as of 2023), and in India alone these data centers have added 1.5 million jobs to the economy over five years, including 169,000 skilled IT positions.[11] We cannot assess the assertion made by Curtis and Avi-Yonah, namely that MAIL did not conduct, manage, or control qualifying exploitation activities to qualify as a CSA participant (as of 2009). From a general transfer pricing perspective, it seems that we would need to focus on analyzing, in a manner similar to how we analyzed MIR, which part of the software development stemming from the activities in Asia, and possibly outsourced by MAIL, are attributable to entrepreneurial functions of MAIL and thus should be considered contributions of MAIL to the CCA.

The third foreign CSA participant is Microsoft Operations Puerto Rico (MOPR), which made a buy-in payment of $17 billion in 2005. According to sources quoted by Curtis and Avi-Yonah, MOPR lacks economic substance and "indeed, a federal district court found that the arrangement

[7] Details on MIR and the other foreign CSA participants are provided in Curtis and Avi-Yonah, *supra* note 3.

[8] The economic substance as such does not appear to be challenged by Curtis and Avi-Yonah. *Id.*

[9] The control over risk functions is an important concept embedded in the OECD transfer pricing guidelines on CCAs. See para. 8.17 and para. 8.15, which say a "CCA participant must have (i) the capability to make decisions to take on, lay off, or decline the risk-bearing opportunity presented by participating in the CCA, and must actually perform that decision-making function and (ii) the capability to make decisions on whether and how to respond to the risks associated with the opportunity, and must actually perform that decision-making function."

[10] Curtis and Avi-Yonah, *supra* note 3.

[11] "Microsoft in Asia: Empowering a New Global Innovation Engine That Is Transforming Economies and Societies," Microsoft (last accessed June 10, 2024).

constituted nothing more than a tax shelter."[12] According to Curtis and Avi-Yonah, as of 2012 MOPR had 177 employees. Considering that MOPR received $1.6 billion in 2012 to build a new manufacturing plant, distribution facility, and product release lab, one can reasonably assume that the local operations and economic substance have been enhanced in subsequent years. Information on the functions performed by MOPR, including entrepreneurial functions, cannot be reliably deduced from the available information. An indication of a substantially limited degree of economic importance of MOPR (compared with MIR and MAIL) can be seen in the fact that MOPR reportedly historically supplied its products only to Microsoft US instead of unrelated customers.

In terms of legacy (that is, previously developed) IP and ongoing development, it is clear that the bulk of contributions to the Microsoft CCA are attributable to Microsoft US. In this context Curtis and Avi-Yonah provide a detailed account of the legacy IP (see below) and indicate that about 29,000 employees located in the region of Washington state produced the programs that were later localized by the other CSA participants.

For the purpose of our analysis, we have established that the CCA has four participants, with Microsoft US having contributed all relevant legacy IP and the other participants later joining with buy-in payments. In addition to the one-time buy-in payments, the foreign CCA participants have also made continuous balancing payments. For general transfer pricing purposes, all CCA participants, perhaps with the exception of MOPR, can be assumed to have sufficient economic substance to — in principle — make valuable contributions to the Microsoft development CCA. To address the main question, namely whether the Microsoft CCA can be considered commensurate with the concept of mutual benefit and thus with the arm's-length principle, we must focus on the actual collaboration on software development.[13]

Development Activities of CCA Participants

Looking at the Microsoft case, the most contentious issue seems to be whether the foreign Microsoft subsidiaries would qualify as participants of the CCA. In this context, Curtis and Avi-Yonah specifically focus on the assertion that "the foreign CSA participant provided no platform contributions or other relevant routine or nonroutine contributions to the CSA."[14] According to the summary of the facts and circumstances provided by Curtis and Avi-Yonah, "Microsoft's foreign CSA participants appeared to play little if any role in the production side of the exploitation activities for the OS [operating system] and app software and primarily performed marketing, sales, and support functions." Further, they assert that the activities of the foreign CCA participants:

> are not an IP development activity that adds to R&D expenses but rather an operating activity akin to packaging or customizing the product for the local market as part of the distribution process. Also, it appears that Microsoft's internal programmers located offshore are performing localization functions that are similar to those performed by unrelated developers using the same tools produced by Microsoft US that are already built into the software.[15]

Considering that Curtis and Avi-Yonah's analysis is embedded in the idiosyncratic framework of IRS regulations on CSAs, we want to emphasize that, for the purpose of our analysis, the framework is instead given by Chapter VIII of the OECD transfer pricing guidelines. Consequently, Curtis and Avi-Yonah's assertion

[12] Curtis and Avi-Yonah, *supra* note 3. According to Curtis and Avi-Yonah, "the ongoing IRS examination of MOPR has revealed that the Puerto Rican entity was created for the exclusive purpose of joining Microsoft's CSA. The IRS has accused Microsoft of engaging in a sham arrangement with Microsoft Puerto Rico."

[13] In our assessment we will put a distinct focus on the ongoing software development contributions of the CCA participants. While Curtis and Avi-Yonah focused more strongly on the exploitation function, we consider the development function to be of more immediate relevance in the context of our article.

[14] *See* Curtis and Avi-Yonah, *supra* note 3.

[15] *Id.* Note that because of their analytical framework, Curtis and Avi-Yonah seem to focus primarily on the exploitation functions, whereas the other development, enhancement, maintenance, and protection functions are not discussed or evaluated in corresponding detail.

would have to be translated to the effect that the contributions of the foreign CCA participants are not unique and valuable. MIR, MAIL, and MOPR should not qualify as participants in the CCA because their participation would be inconsistent with the concept of mutual benefit. As a secondary consideration, if they are allowed to participate in the CCA, one could assert that the buy-in and balancing payments are insufficient to top up the value of their contributions, which are presumed to be of low-value-added, or routine, nature. In this section we address some of the main arguments underlying the assertions made by Curtis and Avi-Yonah and comment from a contemporary perspective on collaborative software engineering.[16]

The viewpoint adopted by Curtis and Avi-Yonah regarding the value of the Microsoft software is reflected in the following statement in their article:

> Windows operates flawlessly for most users with each successive version because the new version consists only of new programming added to the prior versions with a continuous rolling process of development, testing, launch, and patching. In other words, any new version of Windows is basically the previous version with new features layered onto it in a rigorous testing environment.
>
> As a result, the Windows source code has an extremely long decay rate for transfer pricing purposes.[17]

According to this viewpoint, the value attributable to ongoing software development is comparatively low, and the value of previously developed software (embedded in the source code) is comparatively high. Curtis and Avi-Yonah specifically emphasize that undervaluing legacy IP and stipulating artificially low (non-arm's-length) buy-in payments constitutes an important way to shift substantial IP-based profits offshore. Embracing this viewpoint would, however, imply a certain bias, which would possibly undervalue continuous development activities.

Looking at the realities of contemporary software development, the viewpoint adopted by Curtis and Avi-Yonah reflects a too extreme and possibly incomplete interpretation that is not suitable as a default view. The viewpoint is questionable from at least two perspectives.

First, it is questionable whether the Windows OS is indeed only a core of legacy software with layers of new features added. Software developers know that maintaining legacy software is costly, problem-prone, and in need of extra care. For example, older programming languages like C and C++, predominantly used for the development of Windows, are known for being susceptible to vulnerabilities that are based on so-called memory errors. Although there are many approaches for mitigating memory errors in C and C++, those memory issues are still the root cause of about 70 percent of all security vulnerabilities. Modern programming languages inherently protect software from those memory errors and thus from related bugs or vulnerabilities. Although Curtis and Avi-Yonah outline engineering practices that Microsoft uses to address this (such as a rigorous automated testing process), they do not acknowledge other influences on engineering (most notably advances in the hardware that runs Windows OS or changing industry standards). In our experience, those influences will have required Microsoft to perform significant modifications or even a complete reengineering of some subsystems of the Windows OS on multiple occasions.

The second questionable point is that Curtis and Avi-Yonah do not sufficiently differentiate between changes in the actual software (software developers call it the implementation) and the changes in how the software interacts with other systems (interfaces). For example, software applications can store files to a folder using an interface, but the implementation of how an OS technically stores a file has changed significantly over the last decades. Files are no longer stored on floppy disks but rather on hard drives or even faster solid-state drives. The interface applications

[16] We consider the arguments provided by Curtis and Avi-Yonah to be representative of assumptions tax auditors would hold when auditing a CCA on software development. Our comments are thus not to be understood as disagreeing with Curtis and Avi-Yonah on the interpretation of the Microsoft case but rather intended to illustrate that such assumptions may not be applicable to software development CCAs in general.

[17] Curtis and Avi-Yonah, *supra* note 3.

used, however, remain largely the same. The segregation of interfaces (which seldom change) and implementation (which changes often) is a core principle of software engineering. Microsoft, commensurate with this core principle, is known to carefully avoid changes to how Windows OS interacts with the software applications running on it while further developing Windows OS. Every change in how the Windows OS interacts with applications that run on it can potentially create malfunctions of applications that worked seamlessly in an older Windows version. However, keeping these interfaces intact is mostly independent of changing the underlying implementation.[18]

The assumption by Curtis and Avi-Yonah that "any new version of Windows is basically the previous version with new features layered onto it" may be right for interfaces, but it needs to be substantiated for implementations. As a consequence, Curtis and Avi-Yonah might risk overvaluing the legacy IP.

Understanding the nature and organization of the ongoing software development will be crucial when assessing whether the CCA conditions are commensurate with arm's-length conditions. However, for the ongoing software development, likely because of a lack of access to more specific information and data, Curtis and Avi-Yonah merely provide anecdotal evidence by quoting a former Microsoft developer: "Anyone doing technical work at Microsoft only got significant tasks to work on if they were based at Headquarters. . . . Anyone based at any other location were assigned only trivial work."[19] Curtis and Avi-Yonah concede that the anecdotal evidence may be a bit extreme and possibly U.S.-centric, but they generally seem to accept it as reflecting further support for the assertion that the foreign CSA participants did not make valuable contributions.

When interpreted in a more general, contemporary context, the assertion is highly questionable. Software development at an industrial scale is becoming increasingly integrated, collaborative, and self-organized. This manifests, for example, in the industrywide adoption of new software development paradigms, such as DevOps (teams are organized to have capabilities in both developing and operating a software) and InnerSource (teams open their development work to contributions from other teams). Microsoft is a prominent and outspoken proponent of those paradigms. For example, it started adopting InnerSource practices as early as 2008.[20] In a recent talk, Microsoft claimed that 10 percent of work contributions within Microsoft's firm-internal GitHub platform (one of many systems that store Microsoft's software projects) come from teams that are not responsible for the code they are contributing to.[21] In this kind of setup, in which developers routinely pick their own tasks, it seems far-fetched that developers from certain national entities are assigned solely trivial work.

Regarding the localization of the software, which, based on the account provided by Curtis and Avi-Yonah, was one of the main activities performed by the foreign CSA participants, the assumption is also that the activities are routine and contribute only limited value:

> Consider the internet-based platforms through which software is leased, services are provided, and the Microsoft online store reaches customers accessing and paying for products and services. It is understood that the managers and the personnel who operate these platforms day to day are located primarily within the United States. . . . There may be some localization of products and services that occurs outside the United States, but these are mere adjustments to standardized products and services (with the localization options already built in) that are being provided by one central management and operational groups to

[18] In some cases, changing the implementation of a software component but not changing how it interacts with other software applications can become very costly. A well-known example among Windows users is compatibility mode, in which a newer version of Windows OS pretends to be an older version so that applications designed for this older version will run on a newer Windows OS version.

[19] *See* Curtis and Avi-Yonah, *supra* note 3.

[20] Microsoft, "Open Source at Microsoft — CodeBox: Bringing the Open Source Approach In-House," Whitepaper (2008).

[21] *See* Martin Woodward, GitHub, Address at the InnerSource Commons North American Virtual Summit: The Top 5 Myths of InnerSource (Sept. 15, 2020).

Microsoft's customers worldwide through standard platforms.[22]

Whether this assessment holds true depends on how the localization is performed. Are provided functionalities for localization merely used or are the software systems themselves extended?

In terms of the transfer pricing implications, Curtis and Avi-Yonah's assertions effectively translate to disqualifying MIR, MAIL, and MOPR as CCA participants and the finding that the CCA arrangement conflicts with the arm's-length principle. They find that the foreign CCA participants "owned no IP, including legacy IP, at the outset of the new 2009 CSA and made no nonroutine contributions to the CSA at that time that would have provided these affiliates anything more than a 0 percent residual profit split under" the application of a modified residual profit-split method. The hypothetical adjustment of the transfer prices is later calculated by Curtis and Avi-Yonah in allocating a cost-based remuneration to the foreign CCA participants, with 8 percent markup, which they deem "generous" for what they claim to be "clearly routine functions."[23] Curtis and Avi-Yonah present detailed financial data (balance sheets and profit and loss statements) and perform calculations that provide an estimate of the significant amounts of underpaid taxes the IRS auditors could recoup when strictly enforcing tax and transfer pricing laws.[24] Without addressing these calculations in detail, the disqualification of individual entities as legitimate participants in a CCA and the implied reallocation of residual profits would have substantial consequences for any MNE that set up a CCA structure. The key takeaway of the Microsoft case, however, is that the assertions made by Curtis and Avi-Yonah could be deemed circumstantial insofar as they are not based on an evaluation of the collaborative software development actually conducted among the participants of the CCA. Although the data contained in balance sheets or profit and loss statements may provide indirect evidence, the data does not directly reflect the economic substance of the CCA. When aiming to contextualize how much MIR, MAIL, and MOPR participated in an integrated value-creation process for software development in the context of a residual profit-split method (as done by Curtis and Avi-Yonah), a reliable answer can only be obtained through both qualitative and quantitative measurement of contributions to software development by individual parties. The lack of internally (let alone externally) available data on collaborative software development is an obvious obstacle for any such analysis. In the next section we outline a pragmatic and data-driven approach to address the elephant in the room.

Contributions in Collaborative Software Engineering

In analogy to the OECD guidance on profit-splitting factors, it is worth noting that costs (balance sheet data) are generally a "poor measure of the value of intangibles contributed [but] the relative costs incurred by parties may provide a reasonable proxy for the relative value of those contributions where such contributions are similar in nature."[25] Looking at the data and arguments presented by Curtis and Avi-Yonah, including the anecdotal statements from Microsoft employees, it might be inferred that Microsoft establishing MIR, MAIL, and MOPR was largely driven by tax optimization considerations.

In view of the economic substance of these entities (see previous section, "Economic Substance of Foreign CCA Participants"), it would, however, seem that there is no sufficiently reliable proxy available for determining whether these entities are economically legitimate participants in the CCA and whether the buy-in and balancing payments can be considered arm's length. In the next section, we outline three

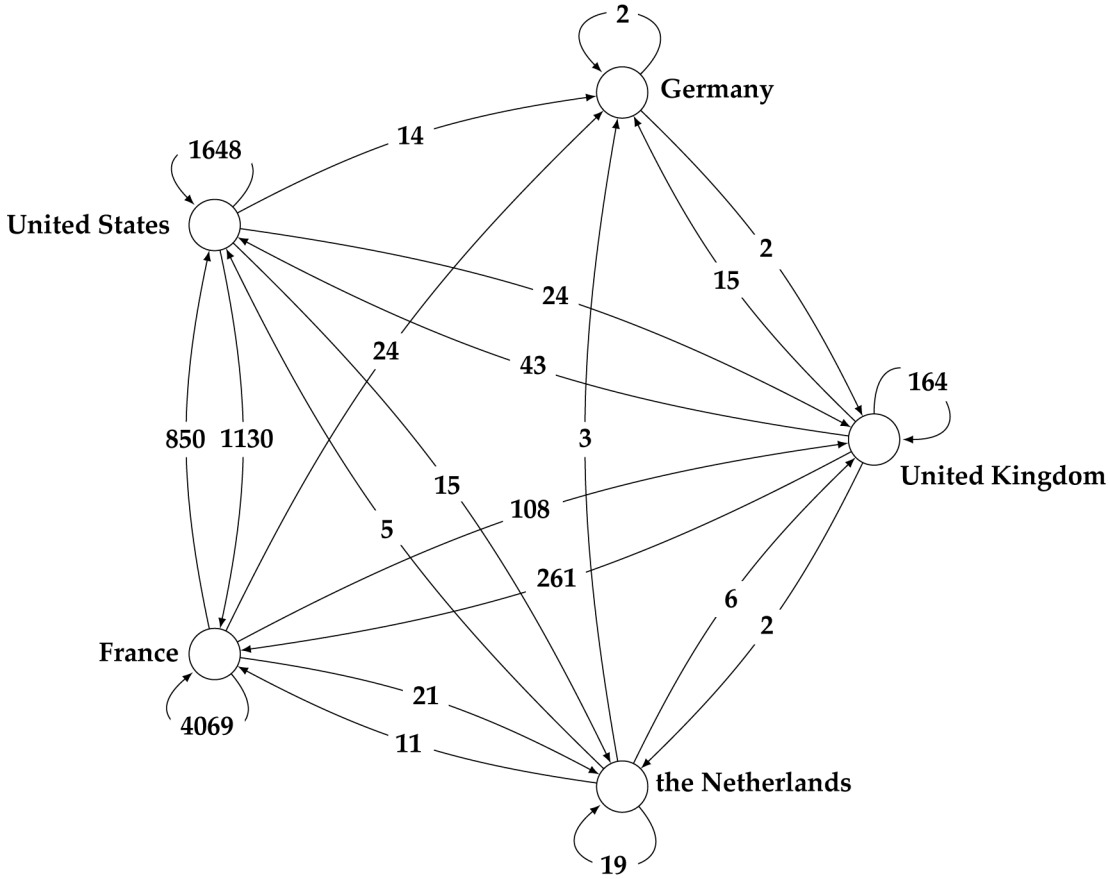
[22] *See* Curtis and Avi-Yonah, *supra* note 3.

[23] *See id.*

[24] Curtis and Avi-Yonah provide detailed calculations of "periodic adjustments." *Id.* Considering that the calculations and corresponding adjustments outlined are highly context-specific (pertaining to U.S. regulations) and therefore do not provide an intuitive basis for general inferences in a contemporary OECD context, such calculations are out of scope of this article. Also, we explicitly do not comment on whether these calculations are a reasonable reflection of possible adjustments under U.S. tax regulations.

[25] OECD, *supra* note 1, para. 2.171.

Figure 1. Visualization of a Dependency Network for Collaborative Software Development



different and complementary approaches from the software development domain to measure and evaluate contributions such as those made by MIR, MAIL, and MOPR. To illustrate the technical properties of these approaches, we use references to real-life data derived from collaborative software development structures in other enterprises to roughly approximate structures like the Microsoft CCA.[26]

Where software development happens, it leaves traces: Each of these approaches uses data that is readily available within the software development infrastructure and tools of most enterprises.

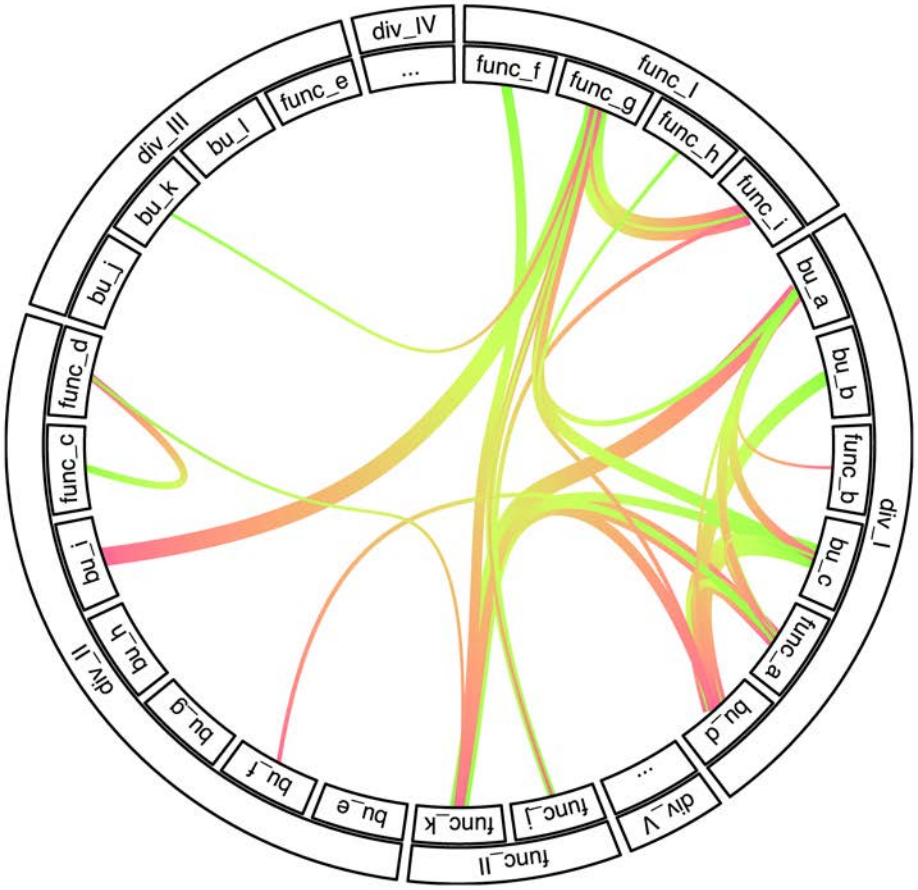
Measuring Dependencies Between Subsystems

Modern software systems are too complex and large for a single developer or team to oversee. Consequently, a software system is typically split into smaller, manageable subsystems. Each subsystem encapsulates a functionality that is conceptually or technologically related. What appears to end users as one large software system is in reality a finely tuned dance of tens, hundreds, or thousands of subsystems interacting. Specialized teams are responsible for the development and maintenance of their subsystems. In an MNE, those teams can be located anywhere worldwide.

If a subsystem interacts with another subsystem, software developers say the subsystems depend on one another. Those dependencies can be automatically extracted from the software development infrastructure

[26]To be clear, we have consulted with neither representatives of Microsoft nor any person who is familiar with the proceedings of the Microsoft case. Any information shown in this section is exclusively based on data analytics derived from different cases, as specified by the references below. No claim is made that the data pattern shown reflects the economic reality of the Microsoft CCA.

Figure 2. Visualization of a Patch-Flow Analysis for Collaborative Software Development



and augmented with information on the national entities responsible for the subsystem.

In a recent empirical study with a large software-driven enterprise,[27] we explored the dependencies among subsystems of a globally distributed software system. We found that about 70 percent of the identifiable dependencies between subsystems cross national boundaries and many legal entities were providing and using subsystems. Figure 1 displays the so-called dependency network or use graph from this software system. The nodes of this graph are anonymized national entities, and the edges denote how many subsystem dependencies exist among the national entities.

In the case of Microsoft, such an analysis could be used to substantiate or invalidate the assumption by Curtis and Avi-Yonah that Microsoft headquarters were the only party supplying the subsystems that constitute Windows OS to the CCA participants.

Measuring the Flow of Code Contributions

Software development has become increasingly collaborative and integrated over the last two decades. While one team might be responsible for a subsystem, it is possible (and probable) that developers also perform fine-grained work contributions to subsystems of other teams (potentially in another country). They do so by proposing or integrating concrete changes to the subsystem's software source code; they perform what is called a code contribution. The flow of these code contributions through an

[27] Michael Dorner et al., "Describing Globally Distributed Software Architectures for Tax Compliance," arXiv:2312.00925 (2023).

enterprise (referred to as a patch-flow[28]) can be automatically measured and analyzed. Figure 2 displays the patch-flow between the divisions and business units of a large automotive supplier. This kind of analysis can easily be performed with national entities instead of divisions and business units. The squares on the outer circle denote divisions (div), business units (bu), and cross-functional teams (func). The patch-flow is denoted by edges flowing within the inner circle).

In the case of Microsoft, this kind of analysis could be used to substantiate or invalidate the assumption by Curtis and Avi-Yonah that employees of the CCA participants are assigned their tasks in a top-down fashion and solely work on trivial work items. The reality of many firms for which we have performed patch-flow analyses is that software developers collaborate at eye level.

An analysis of the patch-flow combined with an analysis of subsystems' dependencies can be a cornerstone of a transfer pricing strategy for collaborative software development. In a recent article,[29] we discussed this using InnerSource software development (one of many software development approaches) as an example.

Measure Communication of Software Developers

During various phases of the software development life cycle, software developers communicate with one another. In globally distributed software engineering efforts, this communication often happens asynchronously and in written form. Specific software tools support developers in communicating, for example, issue trackers (developers report, manage, and discuss defects) or code review tools (developers inspect each other's code and exchange feedback).

The communication relationships, which can be formed among developers from different countries, can be automatically extracted and analyzed. In a recent study we found that up to 30 percent of communication through the proxy of in-code review in a large software-developing enterprise crossed national borders.[30]

In the case of Microsoft, an analysis of communication could augment the other two analyses (dependencies and patch-flow) in substantiating the economic substance of the interaction among CCA participants. The communication flow provides a detailed footprint that enhances the transparency of the integrated nature of the collaboration; the communication flows can also be matched to a functional and risk analysis, albeit requiring additional interpretation regarding the nature of the communication.

Lessons Learned

The Microsoft case shows that tax authorities have a distinct interest in assessing the arm's-length nature of CCAs and other transfer pricing arrangements related to the creation and use of intangibles such as software. The resulting tax risks are high. If tax authorities challenge the setup of the CCA, or the participation of individual entities is not commensurate with the economic conditions and thus does not comply with the arm's-length principle, a reallocation of residual profits — and implied transfer pricing and tax adjustments — is at stake. Even if the arm's-length nature of the CCA is accepted in principle, tax authorities have a clear incentive to challenge the arm's-length nature of buy-in and balancing payments. Solely relying on arguments relating to economic substance and indirect evidence derived from accounting data will be insufficient to determine and defend the economic properties of collaborative software development and how the contributions of individual entities are appropriately balanced.

Considering that many MNEs adopt an increasingly decentralized approach to software development, often involving thousands of software engineers and millions of cross-border code contributions, identifying and delineating the contributions attributable to individual legal entities presents a challenge. Considering that taxpayers are faced with comprehensive

[28] Maximilian Capraro, "Measuring Inner Source Collaboration" (2020) (PhD dissertation at the Friedrich-Alexander University Erlangen-Nürnberg).

[29] *See* Oliver Treidler et al., "Sustaining Arm's Length Cost Allocations for Highly Integrated Development Functions — An Explorative Case Study of Transfer Pricing for InnerSource Communities," *MNE Tax* (2022).

[30] *See* Dorner et al., "Taxing Collaborative Software Engineering," 41(4) *IEEE Software* (2023).

documentation requirements in a post-base erosion and profit-shifting world (especially on IP policies, which would include software, software development, and substance on development, enhancement, maintenance, protection, and exploitation functions), they need to be able to appropriately measure and analyze those contributions. Also, it will be crucial to establish a contractual framework that is consistent with the economic properties of software development.

When looking at the transfer pricing documentation and, ultimately, a tax audit, the narrative presented by the taxpayer must capture the business rationale underlying the collaborative development and be substantiated by reliable and transparent data. While measurements and interpretation will inevitably exhibit a certain degree of fuzziness, the approaches outlined above (measuring dependencies and patch-flow analysis) offer reliable quantitative evidence of how the software collaboration is manifested and constitute a viable approach to mitigating risks such as those faced by Microsoft. Compared with discussions and assessments based merely on balance sheets and profit and loss statements, applying analytical approaches reflecting the actual development activities would — by default — be much more suitable to determine whether the CCA is commensurate with arm's-length conditions. Although technical caveats may surface in the context of data extractions, data availability will not constitute an insurmountable obstacle in the context of software development. Transfer pricing practitioners often encounter difficulties in obtaining sufficiently reliable data, but software developers create a readily available data trail. Any remaining challenges in valuing contributions and balancing payments will be no different and no more challenging than valuations conducted for other intercompany transactions. Although CCAs for software development may initially seem prohibitively complex and thus risky, the availability of measurable data puts taxpayers in a position to ensure and document that their transfer pricing for CCAs is commensurate with arm's-length conditions and avoid Microsoft's situation. ■