The code of law, and the law of code

By Mr Nathanael Lim, Senior Associate, Pinsent Masons MPillay LLP

The paper "*Private International Law Aspects of Smart Derivatives Contracts Utilizing Distributed Ledger Technology*" *[published by ISDA, Clifford Chance, R3 and Singapore Academy of Law dated 13 January 2020*(the "**Paper**") *[*https://go.gov.sg/isda-pte-intl-law*]* sets out an approachable introduction to determining the governing law of derivative contracts which make use of Distributed Ledger Technology ("**DLT**"), in the context of transactions governed by ISDA documentation. Commercial parties and lawyers will be glad to know that the Paper concludes that generally, the English and Singaporean courts will uphold an express choice of law by the contracting parties. This is crucial for deal certainty, as it addresses complications that may arise given the global, cross-border context in which such transactions usually take place in.

The question of governing law is but the first step into the myriad of legal issues that need to be considered in the use of DLT. This article provides brief comments on the Paper before setting out some further thoughts on these issues.

Non-contractual claims; Smart contracts vs Smart Code

It is important to note that the Paper is set in the context of transactions which are ***ultimately governed by a master agreement with an express choice of law provision.*** In the example explored in the Paper, the application of DLT is limited to the more automatable operative provisions of the governing agreement (eg. those relating to payment obligations). In such a setting, it is largely "business-as-usual", and parties should continue to ensure that there are clear and express choice of law provisions in their drafting which will determine the governing law of contractual claims.

In the same parameters, this should not drastically affect the application of traditional principles of conflict of laws to determine the governing law for non-contractual claims as well. Choice of law rules for non-contractual claims are generally premised on identifying a relevant jurisdiction – for example, for a tortious claim where the double actionability rule[1] applies, the jurisdiction in which the tort was committed[2]; for a restitutionary claim arising "outside" of the contract, the jurisdiction where the enrichment occurred[3]. Ultimately, because of the limited scenario in which DLT is used in the transaction, it is still by and large a "conventional" transaction (save for specific portions of the master

[1] *Rickshaw Investments Ltd v Nicolai Baron von Uexkull* [2007] 1 SLR(R) 377

[2] This is determined by the "substance of the tort" test, and the law has developed to cover a wide range of tortious actions in determining where the tort was committed.

[3] See *CIMB Bank Bhd v Dresdner Kleinwort Ltd* [2008] 4 SLR(R) 543. For restitutionary claims arising in connection of a contract, the governing law will be the proper law of the contract (whether expressed or implied).

agreement to which smart code applies), so it is likely that the use of the DLT would not make too significant a difference in identifying a relevant jurisdiction.[4]

What this means though is that as we continue to explore how much further DLT can be incorporated in transactions, parties need to bear in mind that the less "conventional" the transaction is, that being the more "internal" smart contract code is incorporated within a legal document[5], the more this analysis will be tested. The real challenge, which lies outside the scope of this article (and the Paper), is really how traditional conflict of law principles will apply the closer we get to the other end of the spectrum where parties transact entirely on code.

Legal issues surrounding technical "errors"

As pointed out by the Paper, disputes may also arise in relation to any alleged errors in the software code. In the event where the code performs in a way which one party does not expect it to (but perhaps in the reasonable expectation of the other Party), does the aggrieved party have any legal remedy?

A brief case study can illustrate how this may arise, which is the "hacking" of an organisation named the Decentralised Autonomous Organisation ("**DAO**") in 2016. The DAO operated as an automated investment fund, and investments in the fund were represented by the distribution of DAO Tokens. The DAO would use the funds raised to invest in various ventures, and it was described that participants would receive a return on investment in proportion to the DAO Tokens they held. *Inter alia*, the creation of the DAO Tokens (triggered upon an investor sending funds), and the relevant payouts to participants, were governed by code which was made available for viewing to participants.

At the risk of over-simplifying the technical details of the case, an individual (the "**Exploiter**") later "exploited" a part of the code (a recursive call function) which allowed him to extract a "disproportionate" amount of funds from the DAO, in a way that was contrary to what participants of the DAO had understood or expected.

The legality of the Exploiter's actions is complicated because arguably, all the Exploiter had done was to execute commands on a system which had been agreed upon by all Parties, although this resulted in an unexpected and/or undesired outcome. There was no breach of security, no illegal access, and merely the calling of functions of the code to achieve this. Furthermore, language in the "governing" terms and conditions of the DAO seemed to suggest that the applicable code would take precedence over any explanatory terms or description. The matter was eventually resolved privately amongst the relevant stakeholders, so the legal position of the whole matter ultimately remains uncertain.

One seemingly straightforward solution to avoid such a situation would be to have provisions deeming that the written legal agreement should take precedence over software code in an event of conflict. This however, assumes that the written agreement is expressed clearly enough to address the scenario at hand, and that there is an ***obvious*** conflict between the expressed terms and the software code.

[4] This of course, does not address specific issues which may arise from the *type of assets* which are the subject of the transaction. The Paper sets out a brief discussion on the issues that may arise when dealing with non-conventional assets.

[5] For a discussion on an "internal" vs "external" model in relation to applying code to legal agreements, see ISDA and Linklaters LLP, Smart Contracts and Distributed Ledger – A Legal Perspective (August 2017), www.isda.org/a/6EKDE/smart-contractsanddistributed-ledger-a-legal-perspective.pdd

Unfortunately, circumstances are rarely so black and white in nature. In the spectrum of the multiple shades of grey disputes arise in, if it is a matter of contractual interpretation and a quest to determine the amorphous intention of the parties, the incorporation of code in an agreement may in fact affect a determination of this, since the agreed software code is one practical expression of the parties' intent. Another glaring limitation here is that this harks back to the same restrictions in which DLT needs to be applied, ie. that there still is a governing master agreement, and to some extent, a duplicity in effort requiring both drafting of the software code and the written legal provisions which govern the practical effect of the code.

The point here is the same as above – as we look to incorporate more automated or algorithmic code in legal transactions, we need to be aware that traditional legal principles are going to tested. Legal practitioners are going to find themselves in an interesting position if we ever cross the inflexion point one day where code prevails over the written legal agreement (where "code is law"). In fact, some have already started to theorize whether a separate body of law, referred to as *Lex Cryptographia*[6], will be developed to govern such issues.

Some practical considerations

As we continue to push the frontiers of incorporating technology into transactional work, there are a few practical considerations that can be borne in mind:

1. First, an incremental approach to the application of DLT into transactions will be a prudent way forward. Notwithstanding the use of DLT, continuing to have clear and express provisions governing the transaction will give commercial parties the necessary certainty to transact confidently, knowing that in most cases, traditional legal principles can still continue to be applicable.

2. Second, there is going to be a greater interaction between the coding and legal communities. Both sides will need to start to be more comfortable with each other's area of work, and to start to find ways to work together in a productive and efficient manner.

3. Third, the legal community should start to deeply consider and not shy away from new developments in the law needed to tackle with unique issues that may arise from disputes involving evolving technology. In this manner, the decision of the SICC in Quoine Pte Ltd v B2C2 Ltd[7] ("**Quione**") is a good example of how this can be approached. The Quione case traverses multiple areas of law, and the judgment sets out in no small detail the legal nuances that need to be considered when these are applied to evolving commercial technology (in the Quoine case, the use of algorithmic code in determine prices). As jurisprudence continues to develop in this area, lawyers will do well to keep abreast of these changes and keep themselves updated on the exciting new developments this area of law will bring.

[6] See Wright, A. and de Filippi, P. (2015) 'Decentralized blockchain technology and the rise of lex cryptographia', available at < https://www.intgovforum.org/cms/wks2015/uploads/proposal_background_paper/SSRN-id2580664.pdf > (accessed 1 April 2020)

[7] Quoine Pte Ltd v B2C2 Ltd [2020] SGCA(I) 02 https://go.gov.sg/sicc-judgment-2020-sgca-02