

# **Blockchain**

## Smart contracts on the XRP Ledger (XRPL)

**Marcos Sebastián Caso Polo**

**April 2024**

## 1. Introduction to Smart Contracts.

### - What is a Smart Contract?

Smart Contracts as IBM defines them are "digital contracts stored on a blockchain that are automatically executed when predetermined terms and conditions are met"<sup>1</sup>. The importance of Smart Contracts lies in the fact that there is no need for an intermediary to continue with the contract process<sup>2</sup> since the terms and conditions are established in a programming language.

For a Smart Contract to exist the components are: People or entities involved (usually one entity delivers something and the other entity receives something), conditions, actions or activities, programming code, a platform that is where the programming code is going to be written and finally an asset that can be money that is in a financial entity or cryptocurrency, a contract for the sale of a real estate, etc.

Smart Contracts are becoming more and more relevant not only in the world of cryptocurrencies but also in other sectors such as finance, supply chain, real estate and more. It is estimated that in 2022 the market value of Smart Contracts was 1.4 billion dollars and could reach almost 9 billion dollars by 2030 which means a CAGR of 26.5%<sup>3</sup> (In North America alone the market value was 500 million dollars. (Graph 1).

### - How Smart Contracts work.

Smart Contracts work as a chain of actions and reactions where something must occur for a new action to be released. They work under the "if/when...then..." scenario. For all this to happen efficiently all the terms and conditions of the contract must be programmed and only the actors involved can see the results.



For a Smart Contract to work properly, all possible scenarios must be considered and the rules must be clear, for example, money amounts, dates, how transactions or actions will be carried out, how information should be displayed, the actions that will be triggered after

---

<sup>1</sup> IBM: What are smart contracts on blockchain? (<https://www.ibm.com/topics/smart-contracts>)

<sup>2</sup> Santander: What are smart contracts? (<https://www.santander.com/en/stories/smart-contracts>)

<sup>3</sup> Fortune Business Insights: Smart Contract Markets (<https://www.fortunebusinessinsights.com/smart-contracts-market-108635>)

an action has been performed, etc., must be specified. Ultimately all parties involved must be in agreement with what is stipulated in the contract.

Once all the instructions are clear, it must be programmed in a platform such as Ethereum, Binance Smart Chain (BSC), among others, and it will be registered in the blockchain.

- **Applications.**

Smart Contracts guarantee all parties involved that actions will be executed according to the contract codes in a correct way saving time and effort. Smart Contracts can be used in the following cases for example:

Industry	Example
Finance	Exchange of financial assets once the stipulated terms have been fulfilled.
Healthcare	Smart Contracts can be created where only authorized people with some kind of code or password can access private information.
Real State	The property or rental transaction is a good example of the use of Smart Contracts in this type of industry. If the Smart Contract detects payment and verification of ownership then it automatically generates other actions such as transferring the property to a new user.
Supply Chain and Logistics	Payment, shipping and delivery of goods can be guaranteed with Smart Contracts.

- **Benefits of the Smart Contracts.**

Some of the benefits of using Smart Contracts are:

- Transparency and trust.
  - o The parties involved can establish and see all the conditions and rules established in the contract.
- Speed.
  - o Smart Contracts guarantee that once the action has been performed, the subsequent action will automatically be carried out without the need for a third party to review it and thus save time.

- **Autonomy.**
  - With no need for intermediaries, two parties can make money transactions for example without having to wait for the approval of a third party which could be a bank.
- **Accuracy and security.**
  - The actions established in the Smart Contracts will be carried out exactly as written in the contract, which gives certainty and security to the parties involved.

## 2. **XRP Ledger and Hooks.**

### - **Definition of XRP Ledger**

XRP Ledger was launched in June 2012<sup>4</sup> and is a public decentralized Blockchain backed by the company Ripple Labs where you can make transactions in a very fast way, in real time and at a very low cost. Something very important is that its energy consumption is also very low which makes it an environmentally friendly<sup>5</sup> tool and that is because it uses a Ripple Protocol Consensus Algorithm or RPCA instead of the well-known Proof of Work or of Stake which is different from other systems.

The platform can be used for: Payments, Tokenization, NFTs and DeFi (Decentralized Finance)<sup>6</sup> and is an open platform anyone can make use of it. It also has an original cryptocurrency whose name is XRP.

### - **Ripple Protocol Consensus Algorithm.**

Under this protocol XRPL uses a network of independent validators that allow the confirmation of transactions immediately (every 3 or 5 seconds). Unlike the Proof of Work where many miners compete with each other solving mathematical formulas to receive a reward using large amounts of energy, with the Ripple Protocol Consensus Algorithm a group of "Validators" receive the information and approve or disapprove it in a fast way. Currently there are more than 120 validators operated by different institutions such as universities, companies and individuals<sup>7</sup> which guarantees the decentralization of the Blockchain.

---

<sup>4</sup> XRPL: Provide a Better Alternative to Bitcoin: <https://xrpl.org/about/history/>

<sup>5</sup> Binance Academy: ¿Qué es XRP Ledger (XRPL)? : <https://academy.binance.com/es/articles/what-is-xrp-ledger-xrpl>

<sup>6</sup> XRPL: How the XRP Ledger works: <https://xrpl.org/about/>

<sup>7</sup> XRPL: How the XRP Ledger works: <https://xrpl.org/about/>

- **Definition of Hooks.**

Hooks are code snippets that allow developers using XRPL to customize or add tasks or conditions to activities to be developed directly in XRPL. They can be written in any language and then be compiled with WebAssembly<sup>8</sup>. Their importance lies in the fact that they allow greater flexibility and customization in the development of activities in XRPL, which guarantees their correct development.

A basic example of Hook usage could be to receive a notification each time a transaction has been made from one account to another.

### 3. Key Concepts - Escrow and Oracle.

- **Escrow**

Escrow is a security mechanism where XRPL amounts can be blocked until the terms of the contract are fulfilled. It enforces the participation of a third party who will hold the assets until the terms of the contract are fulfilled. Escrow can be by Time, Conditions or both (Chart 2):

- Time: Funds are released after a period of time has elapsed.
- Conditions: Funds are released if certain set conditions are met.
- Combination of time and conditions.

The use of Escrow gives both parties the assurance that the terms of the contracts will be fulfilled without the funds involved being affected.

- **Oracle**

Oracles are third party participants that allow blockchains to access and verify real market information such as currency prices, stock exchange information, whether an action has been performed in the physical world, etc. to complete validate or invalidate Smart contracts. An Oracle does not need to be an entity, it could also be a software.

An example could be a device that thanks to the Internet of Things (IoT) can detect if a merchandise has arrived on time to its destination to send money. In this case the Oracle would be the device that, thanks to its sensors, sends information that the conditions of the contract have already been fulfilled in the physical world.

---

8 Dev: Hooked #1: Smart Contracts on the XRP ledger

: <https://dev.to/wietse/hooked-1-smart-contracts-on-the-xrp-ledger-5eb6>

#### **4. Process of Creating a Smart Contract.**

For the creation of Smart Contracts in XRPL a developer can use Python or Java or JavaScript<sup>9</sup> and it is necessary to set the conditions inside the Hooks. Here are the steps for creating Smart Contracts in XRPL:

- a. Create a wallet with XRP.  
The first step is to create a Wallet that supports the cryptocurrency to be used since this is where the cryptocurrency will be stored.
- b. Import the data.  
Commands must be used to import the information from the wallet code or from different Python codes. In the case of XRPL it is "xrpl-py".
- c. Define the Variables.  
Define the variables to have a better communication management and not to have such extensive names when programming the code.
- d. Define the Hook conditions.  
Define the Hook that will make the condition work. In order to have a very clear Hook the developers must think about time, money, actions, and individuals involved. It should be confirmed if the following will be present:
  - i. An Escrow.
  - ii. An Oracle.
- e. Set up the Hook.  
Integrate the Hook into the smart contract code and test it.
- f. Set up the transaction specifications that will trigger the Hook.  
Specify the types of transactions or events that will trigger the hook to execute.

#### **5. Creating a Smart Contract.**

In this Smart Contract it will be a condition that 1000XRP can only be withdrawn if the current date is less than December 31, 2024. It should be noted that the smallest XRP currency is used for transactions, which is known as "drop". 1 XRP= 1,000,000 drops (Graph 3). Knowledge of Python, the XRPL site and LLM (ChatGPT 3.5) was used to create this code:

##### **- Smart Contract**

```
import datetime
```

```
from xrpl.clients import JsonRpcClient
```

---

<sup>9</sup> XRPL: How the XRP Ledger works: <https://xrpl.org/about/>

```
from xrpl.wallet import Wallet
```

```
from xrpl.transaction import Payment
```

```
def withdraw_if_condition_met(secret_key, destination_address):
```

```
    client = JsonRpcClient("https://s.altnet.ripple.net:51234/")
```

```
    wallet = Wallet(seed=secret_key)
```

```
    account_address = wallet.classic_address
```

```
    current_date = datetime.datetime.now()
```

```
    condition_met = current_date < datetime.datetime(2024, 12, 31)
```

```
    if condition_met:
```

```
        transaction = Payment(
```

```
            account=account_address,
```

```
            destination=destination_address,
```

```
            amount="1000000000", # 1000 XRP
```

```
            sequence=client.get_account_info(account_address)["sequence"],
```

```
        )
```

```
        transaction.sign(wallet)
```

```
        response = client.send(transaction)
```

```
        print("You can have your money")
```

```
    else:
```

```
        print("Cannot withdraw money after December 31, 2024.")
```

## - Resume of the Smart Contract

### 1) Importing commands

You need to import "datetime" from the Python library and "JsonRpcClient", "Wallet", and "Payment" from the xrpl library to interact with the Ripple blockchain (XRPL).

```
import datetime
from xrpl.clients import JsonRpcClient
from xrpl.wallet import Wallet
from xrpl.transaction import Payment
```

- With the Python "datetime" command it will be possible to work with the actual date and time information.
- With JsonRpcClient you can connect to an XRPL node and perform operations such as sending transactions, getting account information, and querying network status using XRP's JSON-RPC API.
- With "Wallet" you can import wallets to handle XRPL.
- With "Payment" it will be possible to perform operations such as payment transactions.

### 2) Defining the variables

```
def withdraw_if_condition_met(secret_key, destination_address):
    client = JsonRpcClient("https://s.altnet.ripple.net:51234/")
    wallet = Wallet(seed=secret_key)
    account_address = wallet.classic_address
    current_date = datetime.datetime.now()
    condition_met = current_date < datetime.datetime(2024, 12, 31)
```

In this part of the transaction the variables are defined, such as the address of the associated account, the date, among others.



### 3) Programming the condition

```
if condition_met:
    transaction = Payment(
        account=account_address,
        destination=destination_address,
        amount="1000000000", # 1000 XRP
        sequence=client.get_account_info(account_address)["sequence"],
    )
    transaction.sign(wallet)
    response = client.send(transaction)
    print("You can have your money")
else:
    print("Cannot withdraw money after December 31, 2024.")
```

We send the transaction to the wallet of the client if the current date is before the 31/12/2024.

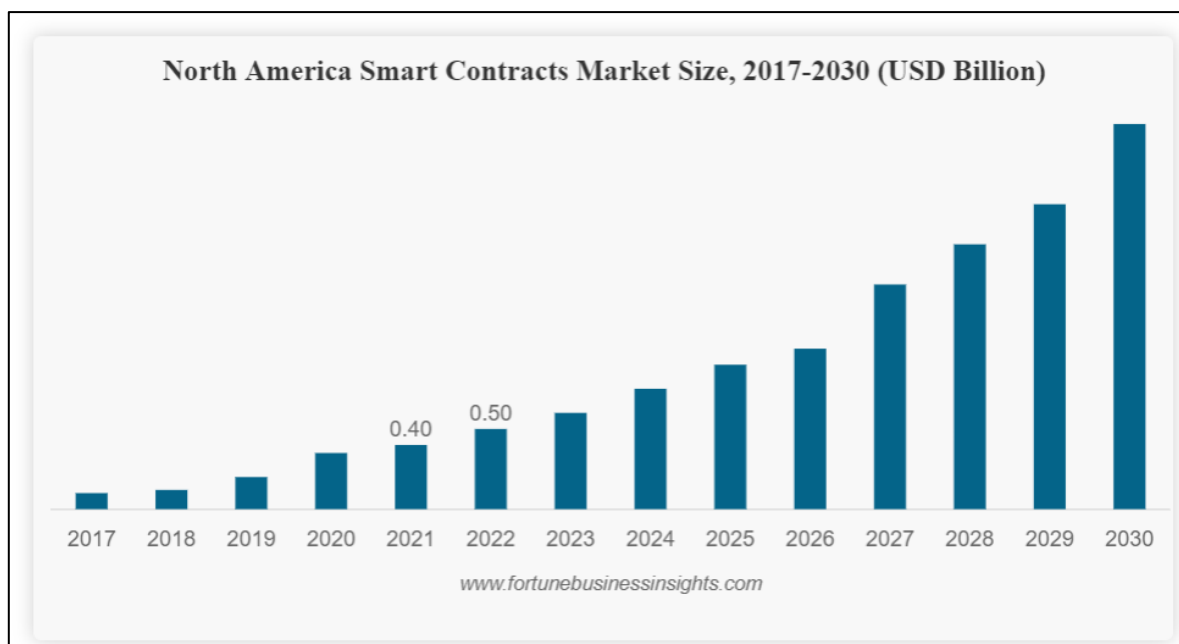
## 6. NFTs on the XRP Ledger

XRP Ledger is a platform that does allow to have NFTs thanks to the XLS-20 functionality that allows the creation and management of NFTs with efficiency and security. Thanks to the XLS-20 standard, transactions can be done in a matter of seconds and with a very low transaction cost.

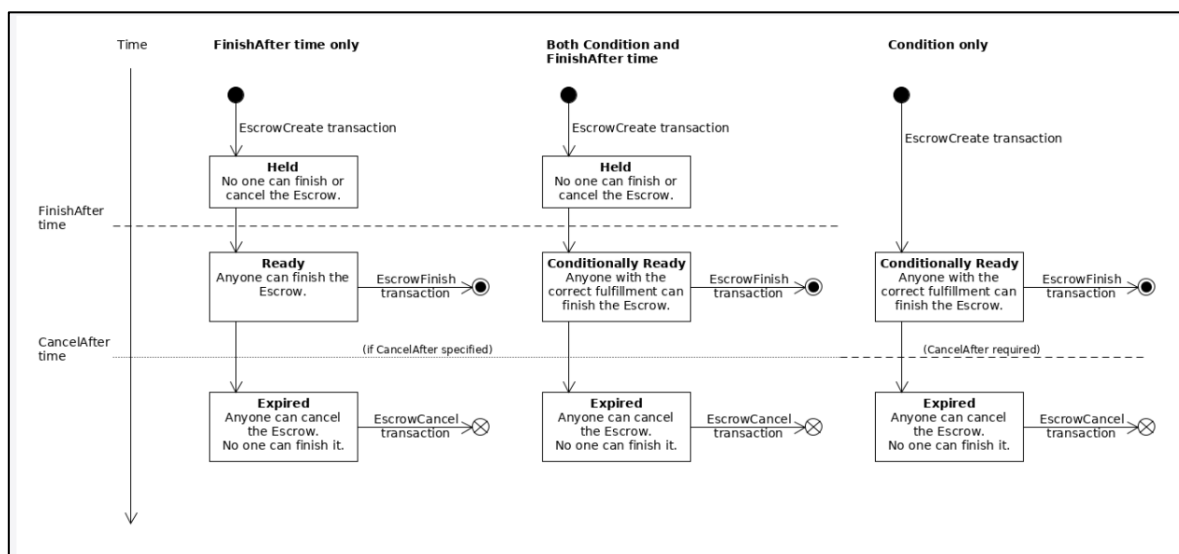
NFTs can be found on sites such as DappRadar (Graph 4) and by October 2023 there were about 3 million NFTs minted in XRP<sup>10</sup>.

---

<sup>10</sup> BeInCrypto: Ripple (XRP) ha acuñado más de 3 millones de NFT con el estándar XLS-20: <https://es.beincrypto.com/ripple-xrp-acunado-3-millones-nft-estandar-xls-20/>

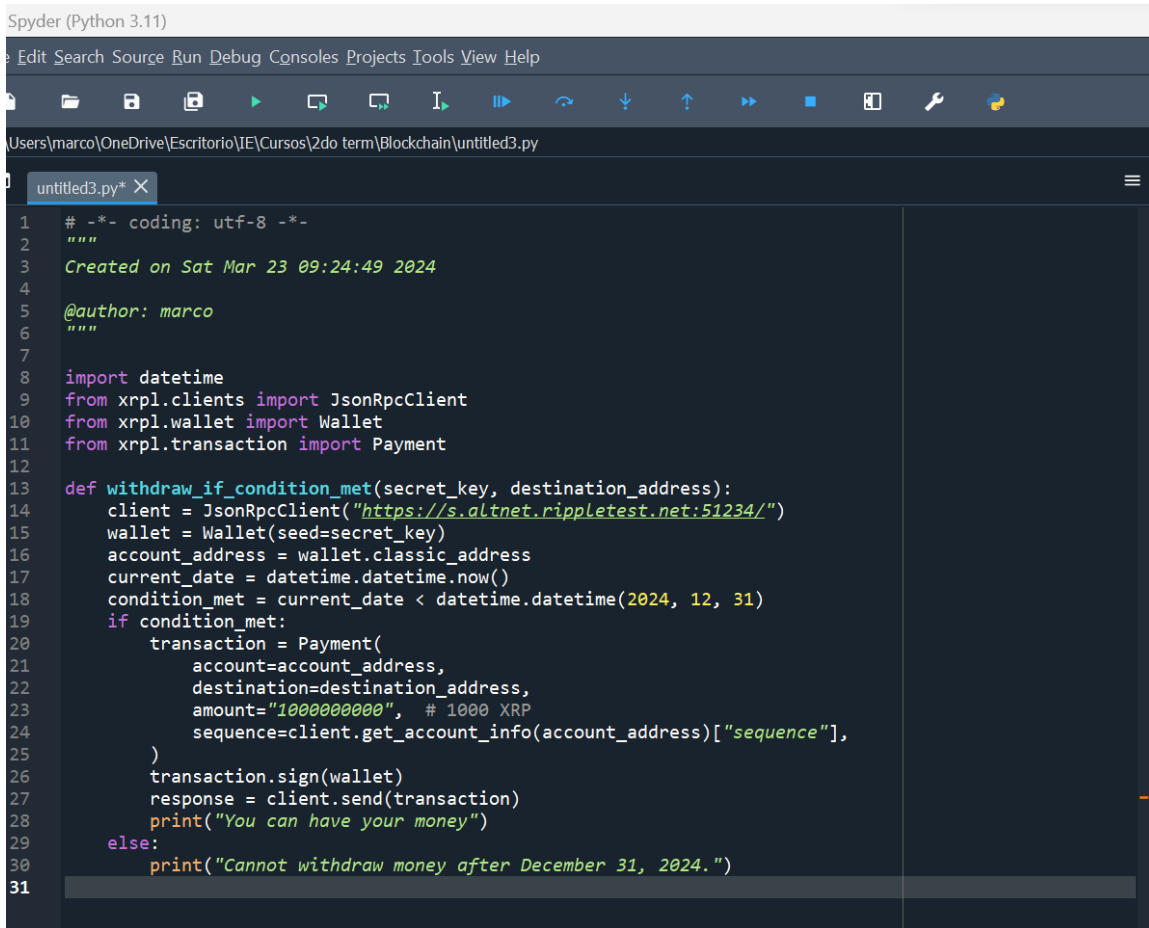
**Graph 1**

Source: Fortune Business Insights.

**Graph 2**

Source: Xrpl.org

### Graph 3



```
1  # -*- coding: utf-8 -*-
2  """
3  Created on Sat Mar 23 09:24:49 2024
4
5  @author: marco
6  """
7
8  import datetime
9  from xrpl.clients import JsonRpcClient
10 from xrpl.wallet import Wallet
11 from xrpl.transaction import Payment
12
13 def withdraw_if_condition_met(secret_key, destination_address):
14     client = JsonRpcClient("https://s.altnet.ripple.net:51234/")
15     wallet = Wallet(seed=secret_key)
16     account_address = wallet.classic_address
17     current_date = datetime.datetime.now()
18     condition_met = current_date < datetime.datetime(2024, 12, 31)
19     if condition_met:
20         transaction = Payment(
21             account=account_address,
22             destination=destination_address,
23             amount="1000000000", # 1000 XRP
24             sequence=client.get_account_info(account_address)["sequence"],
25         )
26         transaction.sign(wallet)
27         response = client.send(transaction)
28         print("You can have your money")
29     else:
30         print("Cannot withdraw money after December 31, 2024.")
31
```

Source: Phyton and LLM.

## Graph 4

#	Collection	Floor price	Avg. price	Mkt Cap	Volume	% Volume	Traders	Sales
1	<b>NEW</b> Anti-Social Media (x) XRP Ledger	-	\$19,83 ^ +100%	-	\$1,86k ^ +100%	+100%	52 ^ +100%	85 ^ +100%
2	XPUNKS (x) XRP Ledger	-	\$418,86 v -8,17%	-	\$1,25k v -85,48%	-85,48%	6 v -72,72%	3 v -84,21%
3	UNIXPUNKS (x) XRP Ledger	-	\$47,97 v -30,7%	-	\$623,73 ^ +197,85%	+197,85%	11 ^ +120%	13 ^ +333,33%
4	Bored Apes XRP Club (x) XRP Ledger	-	\$183,87 ^ +63,59%	-	\$551,62 ^ +63,59%	+63,59%	6 0%	3 0%
5	Pixel Ape Rowboat Club (PARC) (x) XRP Ledger	-	\$12,04 ^ +100%	-	\$377,2 ^ +100%	+100%	15 ^ +100%	29 ^ +100%
6	<b>NEW</b> FUTURE CREATURE (x) XRP Ledger	-	\$29,77 v -29,27%	-	\$327,52 v -22,2%	-22,2%	8 ^ +14,28%	11 ^ +10%
7	DP Camel Collection (x) XRP Ledger	-	\$46,34 ^ +100%	-	\$324,41 ^ +100%	+100%	6 ^ +100%	7 ^ +100%

Source: Daap radar