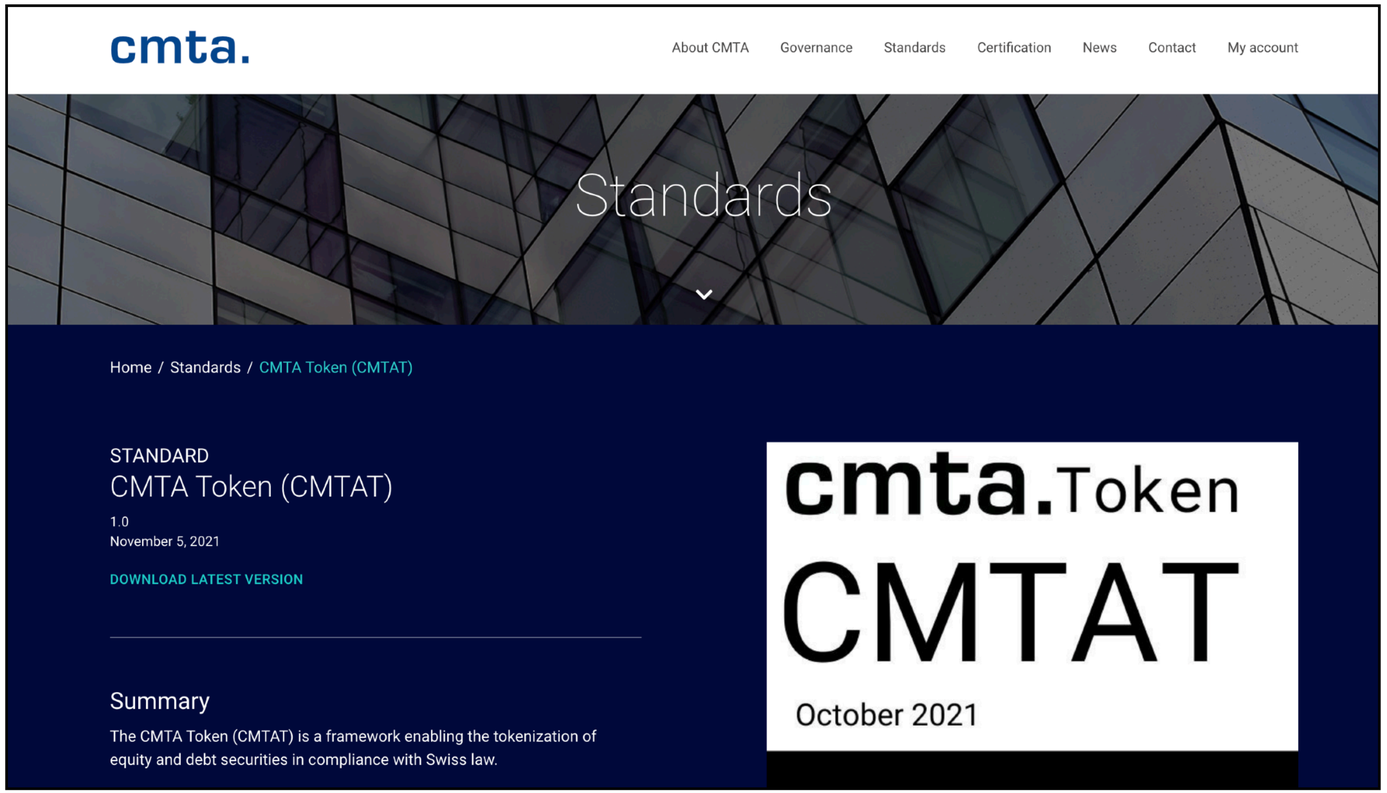
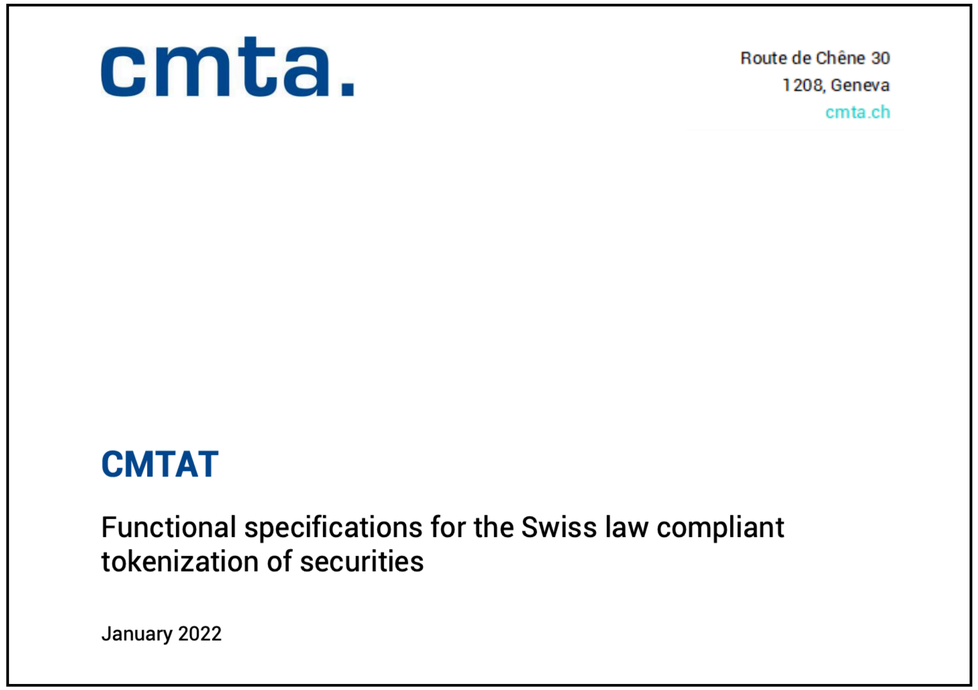
The CMTA Token

JP Aumasson
@veorq

CSO @ taurushq.com

Open standard of a Swiss security token





CMTAT genesis

- **CMTAT** = CMTA Token
- Supersedes the 2019 initial **CMTA20** share token
- **Working group** started in March 2021, with members from: Atpar, Bitcoin Suisse, Blockchain Innovation Group, Hypothekarbank Lenzburg, Lenz & Staehelin, Metaco, Mt Pelerin, SEBA, Swissquote, Sygnum, **Taurus**, Tezos Foundation
  - Definition of requirements and token architecture
  - Reference version and documentation (with external specialists ABDK)

Context: Switzerland' “Lex DLT”

- Officially allows tokenisation of Swiss companies
- Draft in March 2019, full entry into force in **August 2021**
- Introduces the “**DLT Trading Facility**” license
- Treats tokens as “**registered uncertificated securities**"

Art. 58g Minimum requirements for the admission of DLT securities and regular auditing

(Art. 73*d* para. 3 of the FinMIA)

[1] DLT securities may be admitted by the DLT trading facility if the distributed ledger meets at least the requirements under Article 973*d* paragraph 2 of the CO[24].

[2] If the distributed ledger is not operated by the relevant DLT trading facility itself, the facility shall audit the ledger before admitting the relevant DLT securities and regularly thereafter, but at least once a year, for compliance with the requirements under paragraph 1.

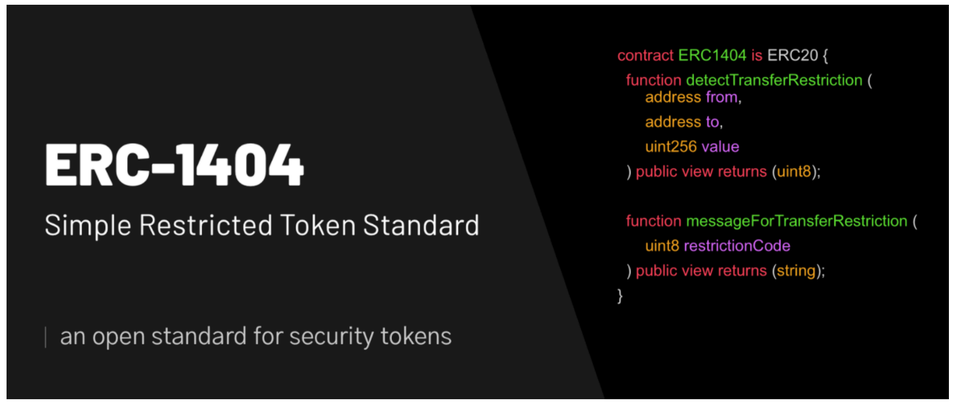
[3] It shall inform its participants of the audits performed and of the findings.

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

https://www.newsd.admin.ch/newsd/message/attachments/67575.pdf

Why not reusing an existing token ?

- Several initiatives (ERC-1400/1404, ST-20, DS, etc.)
- **Not suitable** for CMTA standardisation these:
  - Are often limited to adding **transfer restrictions**
  - Are tied to a **specific** blockchain or ecosystem
  - Are not designed to conform to the **Swiss law**



Functionalities

- **Modular** design, with mandatory and optional modules
  - **Base** (mandatory): basic transferable token functionalities

1. **TotalSupply**: For a particular CMTAT token, any person may know the total number of tokens in circulation at any point it time. ●
2. **BalanceOf:** For a particular CMTAT token and a particular user, any person may know the number of tokens currently recorded on the user's ledger address. ●
3. **Transfer**: Users may transfer some or all of their tokens to some other ledger address (that the transferor does not necessarily control). ●
4. **Mint:** Issue a given number of tokens to a given ledger address. ■
5. **Burn:** Burn (destroy) a given number of tokens from a given ledger address. ■
6. **Pause:** Prevent all transfers of tokens on the ledger until **"Unpause"** is called. ■
7. **Unpause:** Restore the possibility to transfer tokens on the ledger, in principle after **"Pause"** is called. ■
8. **Kill:** Self-destruction of the contract and effectively of the tokens, thereby preventing any transfer or other operation. ■

Mandatory attributes, applicable to all CMTAT tokens:

- Name
- Ticker symbol (optional)
- Token ID (ISIN or other identifier) (optional)
- Reference to the terms of tokenization, the terms of the instrument, and other relevant documents (e.g. prospectus or key information document). The reference can take the form of an URL, a combination of an URL and of specific directions allowing the user to retrieve the relevant documents (e.g. "[domain].com/shares > Tokens") or a fingerprint.

Functionalities

- **Modular** design, with mandatory and optional modules
  - **Base** (mandatory): basic transferable token functionalities

1. **TotalSupply**: For a particular CMTAT token, any person may know the total number of tokens in circulation at any point it time. ●
2. **BalanceOf:** For a particular CMTAT token and a particular user, any person may know the number of tokens currently recorded on the user's ledger address. ●
3. **Transfer**: Users may transfer some or all of their tokens to some other ledger address (that the transferor does not necessarily control). ●
4. **Mint:** Issue a given number of tokens to a given ledger address. ■
5. **Burn:** Burn (destroy) a given number of tokens from a given ledger address. ■
6. **Pause:** Prevent all transfers of tokens on the ledger until **"Unpause"** is called. ■
7. **Unpause:** Restore the possibility to transfer tokens on the ledger, in principle after **"Pause"** is called. ■
8. **Kill:** Self-destruction of the contract and effectively of the tokens, thereby preventing any transfer or other operation. ■

Optional attributes, applicable to tokens used for debt securities:

- Guarantor identifier (if any)
- Bondholder representative identifier (if any)
- Maturity date
- Interest rate
- Par value (principal amount)
- Interest schedule format (if any). The purpose of the interest schedule is to set, in the parameters of the smart contract, the dates on which the interest payments accrue.
  - Format A: start date/end date/period
  - Format B: start date/end date/day of period (*e.g.* quarter or year)
  - Format C: date 1/date 2/date 3/….
- Interest payment date (if different from the date on which the interest payment accrues):
  - Format A: period (indicating the period between the accrual date for the interest payment and the date on which the payment is scheduled to be made)
  - Format B: specific date
- Day count convention
- Business day convention
- Public holidays calendar

Functionalities

- **Modular** design, with mandatory and optional modules
  - **Snapshot** (mandatory): basic transferable token functionalities

1. **ScheduleSnapshot**: For a particular CMTAT token, the issuer may schedule the creation of a snapshot at a certain time. The time of the newly scheduled snapshot cannot be before the time of the latest scheduled, but not yet created, snapshot.
2. **RescheduleSnapshot:** The issuer can change the time of a scheduled snapshot. The new scheduled time cannot be before the time of the previously scheduled snapshot or after the time of the next scheduled snapshot (*i.e.* scheduled snapshots cannot be reordered).
3. **UnscheduleSnapshot**: For a particular scheduled snapshot, the issuer can cancel a previously scheduled snapshot. The unscheduled snapshot must be the last scheduled snapshot, and its time must be in the future.

4. **SnapshotTime**: For a particular scheduled, but not yet created, snapshot, anyone may know the snapshot time.
5. **SnapshotTotalSupply**: For a particular created snapshot, anyone may know the total number of tokens that were in circulation at the snapshot creation time.
6. **SnapshotBalanceOf**: For a particular created snapshot and a particular ledger address, anyone may know the number of tokens recorded on the relevant ledger address at the snapshot creation time.

Functionalities

- **Modular** design, with mandatory and optional modules
  - **Validation**, **Authorization** (optional, legally)

Functionalities:

1. **ValidateTransfer**: Send a request for validation of a particular transfer, given the sender and recipient addresses, and the amount to the issuer.
2. **SetRuleEngine**: Assign a set of rules to be enforced by the **ValidateTransfer** function. Said rules are to be defined in a separate contract.

d. Authorization module

Rationale: Issuers may wish to implement a role-based access control to the token functionalities, rather than distinguish only between issuer and user. This may help reflect the issuer organization's governance model. The Authorization module thus allows the issuer to assign responsibilities and authorizations to various persons (accounts).

Functionalities:

1. **GrantRole**: Grant a role to a given account.
2. **RevokeRole**: Revoke a role from the given account.
3. **HasRole**: Tell whether a given account has a given role.

Corporate actions management

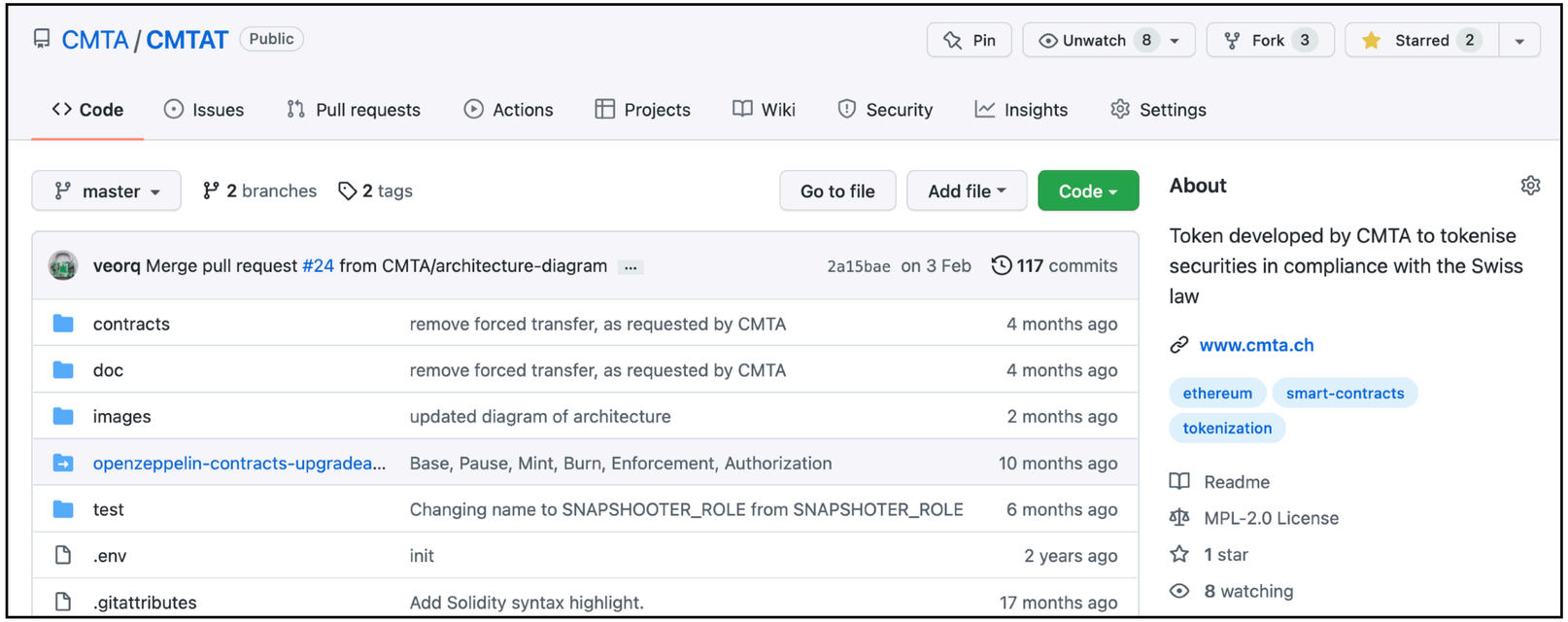
- **Guidelines** on how to deal with:
  - Split / reverse split: can deploy a new token contract
  - Distribution of divided or interest: can be triggered from a snapshot
  - Credit events: API defined to set flag/rating of an account

A module to manage on-chain credit events may support the following functionalities:

1. **FlagDefault**: The tokens are flagged as representing debt instruments in respect of which the issuer has defaulted (bond agent function). ■
2. **FlagDefaultRemove**: The tokens are no longer flagged as representing debt instruments in respect of which the issuer has defaulted (bond agent function). ■
3. **FlagRedeemed**: The tokens are flagged as representing debt instruments that have been redeemed (bond agent function).■
4. **SetRating**: The tokens are flagged as representing debt instruments that have been given a particular rating (bond agent). ■

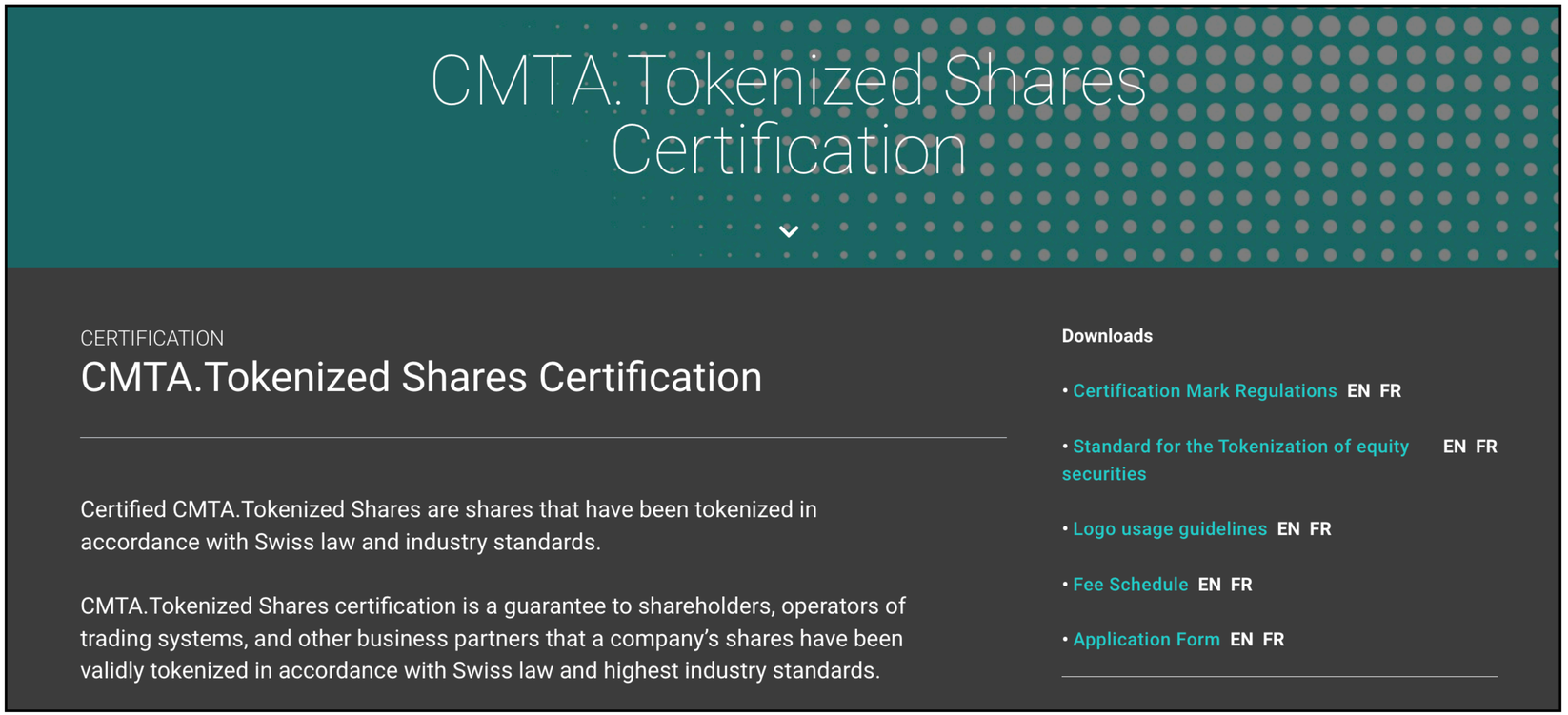
Reference implementation

- Open-source **Solidity** version at https://github.com/CMTA/CMTAT/
- Supports **proxying and gasless** transactions with standard APIs
- Security audit by ABDK



Certification process

- Process to assert that a token is conform with the law (e.g. CMTAT extensions, variants)
- Recognized experts for **legal** and **tech** aspects appointed by CMTA



Conclusions

- CMTAT will **evolve**: debt, structured products, other platforms and protocols
- CMTAT is **integrated** in issuance platforms
- Several companies **used** CMTAT to tokenise their shares
- More at
  - https://cmta.ch/standards/cmta-token-cmtat
  - https://cmta.ch/certification/cmta-tokenized-shares-certification

Thank you!

JP Aumasson
@veorq

CSO @ taurushq.com