KxcRefcode: CKAK02TSCAWH

Tezos TSCA White Paper

on the conceptualization and prototyping of TSCA—Typical Smart Contract Agency

HAOCHEN M. KOTOI-XIE and THE TSCA PROTOTYPING TEAM

We introduce a novel idea related to the Tezos blockchain technology—TSCA, which is the short for "Typical Smart Contract Agency".

TSCA is the idea of having an Agency service available as a webapp where average Tezos users can visit to create smart contracts from predefined templates and interact with them without having to know a lot of technical details about the Tezos blockchain.

We have successfully tested and demonstrated the validity of this idea by implementing a prototype Agency and developing a couple of contract templates.

1 WHY TSCA

The Tezos blockchain is a powerful platform for smart contract development and applications, but without a service like TSCA, it is difficult for the average user to leverage such platform capabilities.

Without TSCA, in order to utilize smart contracts, a user needs to either have the skills to program and operate his/her own smart contracts, which is very difficult; or to use a full-blown 3rd party service that is built upon the Tezos smart contract technology, which is usually very restrictive.

TSCA provides something in-between, where the users could create contracts customized to their needs/tastes, without needing to become overly hacky, given a suitable smart contract template being available.

TSCA may also culture a community-driven ecosystem where template vendors focus on developing high quality contracts, while the Agency handles the tedious part of making those contracts available and accessible.

2 THE PROTOTYPING PROJECT

We, the TSCA Prototyping Team has successfully developed a prototype to demonstrate the viability of the core concepts and technical feasibility of a Typical Smart Contract Agency on the Tezos blockchain.

More specifically, we set and achieved our objectives to the following for the prototyping project:

- define, explain, and sets the definition and expectations for the concept of TSCA
- implement and make open-source a *Software Prototype* that realizes the basic ideas around the concept of TSCA
- launch and operate a *Demonstration Service* that demonstrates the basic ideas and capability of the concept
- develop a few high-quality and useful *Smart Contract Template* and make it available on the demonstration service

In the following sections, we describe the TSCA prototype we have implemented from a technical perspective. For an overview from a user's perspective, please see Fig. 1 for a brief introduction.

This is a registered document in the KXC Technical Note Archives.

Authors' address: Haochen M. Kotoi-Xie, haochenx@acm.org; the TSCA Prototyping Team
Full reference code: HXRD.CKAK02TSCAWH~Ver01. Document rendered at 2022-04-14 23:47 JST.

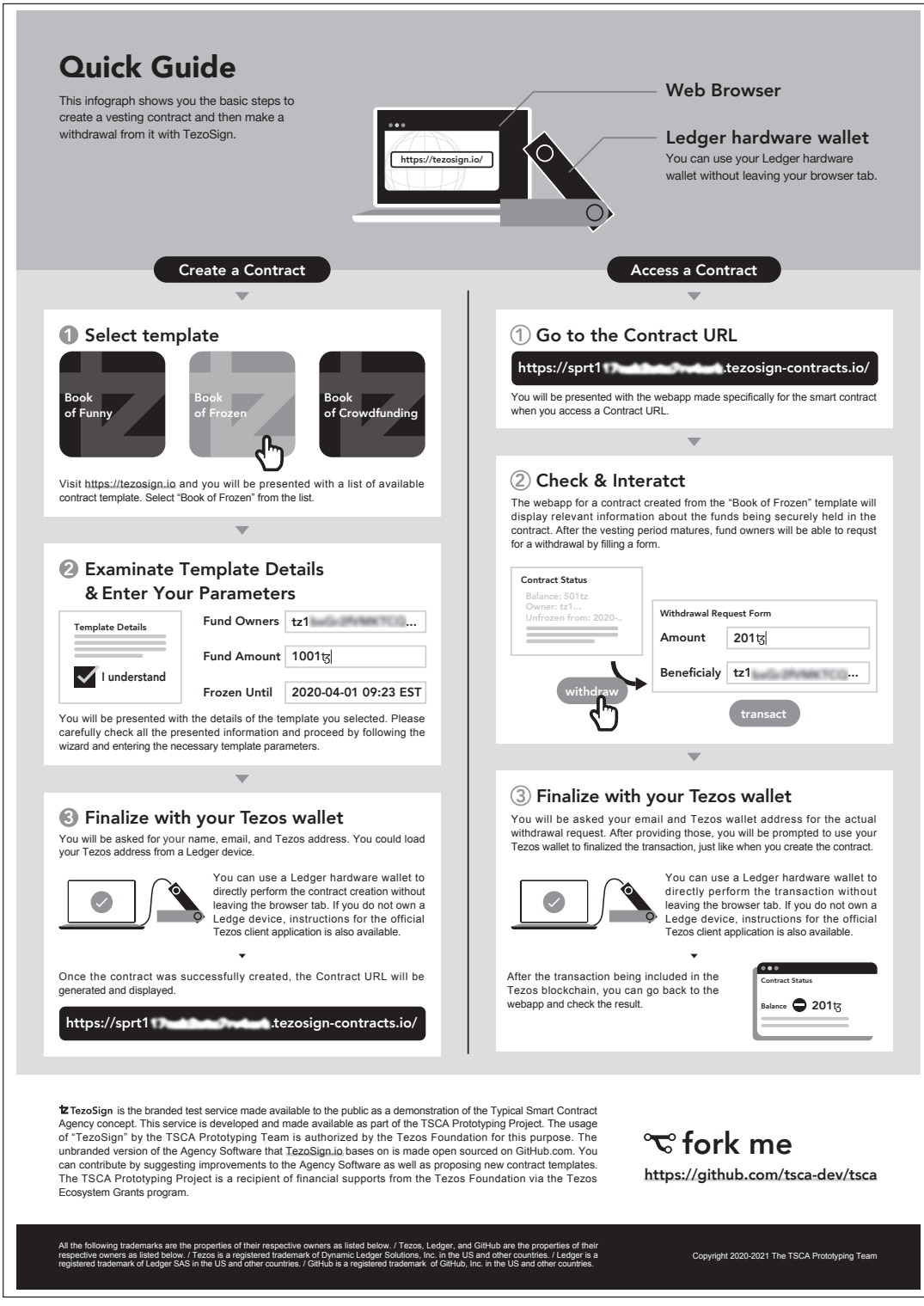

Fig. 1. TSCA Prototype User Perspective

3 ARCHITECTURE OF TSCA

To implement the features expected for TSCA, we have designed the following software/conceptual architecture for the Agency and the Templates to be offered on the Agency.

3.1 Terminology & Components

To help avoiding confusion in terminologies, we have made the following terms in referring different components and concepts.

- *Template Book* refers to a smart contract template with additional information and technical/UI components as a single deployment unit to the agency. each book is identified by its "*bookhash*"
- *Agency and Bookapps* - when used in referring to UI components, the former refers to the primary interface of an TSCA implementation, which lists and allows contract origination from templates; and the latter refers to the interface each template provides to interact with contract instances originated from that template. each bookapp is identified by the template book supplying it
- *Genesis and Origination* the action where a user creates a smart contract from a template book is termed "genesis" or "origination", and we also say "a contract is originated from a template book"
- *Spirit, Ensemble, and Avatars* - the first term refers to a conceptual smart contract instance originated from a template book. yet since a conceptual contract may need more than one actual smart contacts deployed to the Tezos blockchain to implement all its features, we refer to each of the actual/physical on-chain contract as an "*avatar*", to a collection of those avatars that together implements a conceptual contract as an "*ensemble*", and to the conceptual instance of smart contract as, again a "*spirit*". each spirit is identified by its "*sprthash*" and each avatar within an ensemble is identified by its "*rclabel*"
- *Spell and On-Chain Parameter* - due to the conceptual separation of a "spirit" and a physical on-chain contract, there arises a need for terms to differentiate parameters to them. "*spell*" is used to refer to parameter to/for a spirit, and "*on-chain parameter*" is used to refer to the actual Michelson argument in invoking a physical smart contract. there are also a few terms directly related to "spells":
  - *Spell Interpreter* - obviously, a spell needs to be interpreted (i.e. converted) into on-chain parameters in order to forge an actual Tezos operation. to perform such interpretation, each template book is required to include a technical component named *spell interpreter*
  - *Spell Assistant* - another motivation for this separation is to allow spells being expressed not in Michelson, but in a more user-facing data format, such as JSON or a format chosen by the template vendor. yet the user is not expected to type in spells directly, and each template book is required to include UI components named *spell assistants* to help the user formulating a spell each spell assistant is identified by its "*salabel*" and the template book that supplies it
  - *Spell of Genesis* refers to the spell, i.e. the parameters, to a template that parameterize the smart contract to be originated. a Spell of Genesis is typically formulated as part of contract origination process
  - *Spell to Spirit* refers to the spell, i.e. the parameters, to a spirit in order to interact with it in a parameterizable way. a Spell to Spirit is typically formulated as part of a contract interaction session where the user visit the Bookapp for a spirit and perform some operation regarding it

The software implementation is divided into the following components.

- *Broker Contract* is the physical smart contract controlled by the operator of a TSCA compatible service (simply "TSCA operator" in the remaining of this document) that handles actual on-chain contract origination and fee collection.
- *Chain Indexer* is the program operated by the TSCA operator that collects real-time information from the Tezos blockchain about the Broker contract and the avatars originated from that Broker contract, for allowing the Agency and Bookapps to produce correct display of information.
- *Agency Top* is the top-page of the Agency webapp. the Agency Top is tasked to
  - list available template books
  - allow users to select/locate a template book and navigate to its Book Top
  - allow users to navigate to the bookapp for a spirit
- *Book Top* is the top-page for a template book. the Book Top is tasked to
  - display basic and detailed information about the specific template book, in order to help users decide whether to originate a contract from this book
  - in the future, allow book-specific branding to some extend to promote this book
  - allow users to start the process of originating a contract from this book
- *Spell Assistant Presenter* is the components of the Agency webapp that renders a spell assistant supplied by a template book in the browser
- *Chain Clerk* is the components of the Agency webapp that takes a spell (as well as which book/spirit it is for) and collects necessary additional information[1] to forge a Tezos transaction, then forges and simulates the transaction and provides instructions to the user to actually inject the transaction to the Tezos blockchain

3.2 Template Book & Bookapp

A Template Book is the deployment unit of smart contract templates to a TSCA compatible service. Each Book is required to supply the following components.

- *Book Details* that will be displayed to the user to help selecting which template to use
  - *Basic Information* - for example the book title, a synopsis, the template vendor, the template usage fees, etc.
  - *Contract Parameters Listing* that lists and explains all parameters the contract template expects/supports
  - *Contract Terms in English* that describe the behavior and effects of the smart contract being originated from this template in plain English, much like articles in traditional contracts
  - *Contract Caveats* list vendor specified caveats the user should be aware of before using this template
  - *Formal Contract Specification* gives technical description (like source code) and formal discussion (like technical reports containing proof of correctness) of the behavior of smart contracts being originated from this template
- *Spell Interpreter* that interpret spells (including spells of genesis and spells to spirits) into on-chain parameters
- *Spell Assistants* that are wizards to guide users to provide necessary information and formulate a corresponding spell
- *Ensemble Generator* that when taking an on-chain parameter interpreted from a spell of genesis, generates the actual Michelson code for each avatar and the initialization invocations for the ensemble in accomplishing a genesis request. the generator must be a Michelson program and will be executed by the Broker contract

[1] including technical information like source address, and legal information like the identity and contact of the user

- *Bookapp* is a webapp supplied by the template vendor that serves as the user interface for a spirit originated from the same template. a bookapp is required to be packed as an SPA[2] with one HTML file as the entrypoint and as many JavaScript/image/etc. assets as necessary, but is forbidden to access the internet except via certain APIs. a bookapp, via a set of well defined APIs provided by the hosting Agency webapp, will be able to
  - access information like storage, balance, and transaction history about each of the avatar in the spirit ensemble
  - request the display of a Spell Assistant with certain information pre-filled. see slide **??** for more information about user interaction in general with the blockchain in general

3.3 Spell Assistant & Chain Clerk

Under the architecture of TSCA, in order to interact with the Tezos blockchain (e.g. to originate a contract or to inject an invocation of a spirit), the Book Top or Bookapp of a template book must display a Spell Assistant and from there, Chain Clerk will be presented to the user in helping inject an actual transaction to the Tezos blockchain. This workflow is summarized below.

(1) the user is prompted to perform an on-chain action
(2) the Book Top/Bookapp request the Agency webapp to display a certain Spell Assistant corresponding to the action being prompted
(3) the Agency webapp will pop-open the Spell Assistant Presenter, and render the Spell Assistant being requested
(4) the user, following guidance of the Spell Assistant, fill in all the required information asked by the Spell Assistant and proceed to the Chain Clerk
(5) receiving the spell formulated by the Spell Assistant, the Agency will display the Chain Clerk and prompt the user for any additional necessary information
(6) the user, prompted by the Chain Clerk, input those information
(7) the Chain Clerk then invoke the Spell Interpreter of the template book, and together with the additional information, forges a Tezos transaction, then simulates and displays the simulation result as well as instructions for transaction injection to the user
(8) if all look good, the user can proceed and follow the instructions to actually inject the transaction to the Tezos blockchain

such an architectural design provides template developers an abstraction over interactions with the actual blockchain, and provide clear clues to users about exactly when, and exactly what content of an on-chain transaction is going to be injected to the blockchain.

3.4 Broker Contract & Avatar Wrapper

Under the architecture of TSCA, contracts origination from a template book is performed indirectly via an invocation of the Broker Contract, and every physical contract originated by the Broker Contract is wrapped with the Avatar Wrapper.

What happens when a contract is originated is summarized below.

(1) the user formulate the spell of genesis with the help of a spell assistant
(2) the spell of genesis is interpreted into on-chain genesis parameter by the spell interpreter of the corresponding template book
(3) the Tezos transaction the chain clerk forge will instruct the Broker contract to execute the Ensemble Generator of the corresponding template book while passing the genesis parameter

[2]Single-Page Application

(4) the Ensemble Generator will generate the code, initial balance, initial storage, and rclabel for each avatar in the to-be-originated ensemble, as well as an initialization invocation sequence
(5) the Broker contract, after executing the Ensemble Generator, will originate the ensemble and perform the initialization invocations as per specified by the executing result of the ensemble generator, where each avatar in the ensemble is wrapped with the Avatar Wrapper
(6) if everything above was successful, the Broker contract will save the fees collected from the user and update the corresponding accounts for each fee component

The decision to wrap every avatar with the Avatar Wrapper in TSCA is due to the technical limitation of the CREATE_CONTRACT instruction of the Michelson language, which only allows a contract being originated with a program hard coded into the source code of the Broker contract.

Yet even without the technical limitation mention above, it is still beneficial to wrap avatars since it allows additional features like the ones listed below to be implemented, although it comes at the cost of increased contract code size, and thus network fees to users.

- to allow fee collection per spirit invocation, in addition/in place to fee collection per origination
- to allow AML/KYC restrictions being enforced as a service provided by the Agency
- to provide an Agency related contextual API/library to the wrapped contract

4 USER EXPERIENCE DESIGN

The user flow for contract origination and interaction is summarized below.

4.1 Contract Origination

This user flow assumes that the user has not decided which template to use.

(1) the user visit the Agency Top
(2) the user is presented with a list of available template books, where each book is listed with its title, synopsis, and usage fees
(3) the user browse the list of books, and click on one that interests him/her
(4) the Book Top for the clicked book is displayed
(5) the user examine the Book Details to learn more about the book
(6) at this point, the user could click the Back button of the browser to go back to the Agency Top and click on another book listing to examine its Book Details
(7) the user repeat Step 3–6 until he/she finds a template book that he/she wants to originate a contract from
(8) the user carefully examine all the details provided on the Book Top, ticking all the "I understand" confirmation checkboxes following every "Contract Term" and "Caveat" entry, and then click the "Launch" button to initialize the origination process
(9) a pop-up will be displayed showing the "Terms of Service", after agreeing to those by ticking the checkboxes, the user clicks on the "Proceed" button to proceed
(10) the Agency will display the Spell Assistant for the Spell of Genesis of the template book the user selected. this assistant will guide the user to input all the template parameters
(11) the user follows the guidance of the Spell Assistant and input all the required parameters, then click "Proceed" button to proceed
(12) the Chain Clerk will be presented, where
  (a) the user will be asked for his/her name, email, and Tezos address. the user could load his/her Tezos address from a Ledger device
  (b) first time user or returning user using a new Tezos address will receive a confirmation email with a link to click to finish the confirmation process

(c) after the user inputting those information and clicking “Proceed”, Chain Clerk will forge the Tezos transaction, perform simulation for applying that transaction, and display the simulation results, including network/template usage fees to the user
(d) after the user examine and accept the simulation result, he/she ticks the “I understand” confirmation checkbox
(e) when the confirmation checkbox is ticked, instructions to inject the forged operation to Tezos network are shown. the user also has the option to proceed with a Ledger device
(f) at the same time, URL to the Bookapp for the contract to-be-originated will also be shown
(g) the instructions, the Bookapp URL, and a description of the origination request will also be emailed to the user’s email address
(h) the smart contract will be originated after the user successfully following the prompted instructions
(i) the user can access the bookapp URL to check contract status and further interact with it

4.2 Bookapp Interaction

This user flow assumes that the user knows the Bookapp URL to the contract originated from a template.

(1) the user visit the Bookapp URL
(2) the Bookapp will display the status of the smart contract and available interactions with it
(3) the user examine the status of the contract and decide to perform a certain interaction with it
(4) the user click on the button associated with the interaction he/she wants to perform
(5) a pop-up will be displayed showing the “Terms of Service”, after agreeing to those by ticking the checkboxes, the user clicks on the “Proceed” button to proceed
(6) the Agency will display the Spell Assistant for the selected type of interaction. this assistant will guide the user to input all the parameters needed to perform the interaction
(7) the user follows the guidance of the Spell Assistant and input all the required parameters, then click “Proceed” button to proceed
(8) the Chain Clerk will be presented, where
  (a) the user will be asked for his/her name, email, and Tezos address. the user could load his/her Tezos address from a Ledger device
  (b) first time user or returning user using a new Tezos address will receive a confirmation email with a link to click to finish the confirmation process
  (c) after the user inputting those information and clicking “Proceed”, Chain Clerk will forge the Tezos transaction, perform simulation for applying that transaction, and display the simulation results, including network fees to the user
  (d) after the user examine and accept the simulation result, he/she ticks the “I understand” confirmation checkbox
  (e) when the confirmation checkbox is ticked, instructions to inject the forged operation to Tezos network are shown. the user also has the option to proceed with a Ledger device
  (f) the instructions together with a description of the interaction will also be emailed to the user’s email address
  (g) the interaction to the smart contract will be performed on-chain after the user successfully following the prompted instructions
  (h) the user can go back to the Bookapp to check the effects of the interaction he/she just performed

5 DEMO SERVICE: TEZOSIGN.IO

To help conveying the concept and capabilities of TSCA, the project team plans to launch and operate a demonstration service that runs a branded version of TSCA/p1. The branding name is chosen to be "*TezoSign*", taking from the action "*Signing* a smart contract on the *Tezos* network".

The demonstration service is launched at https://alpha.tezosign.io and users could visit to originate and interact with contracts on the Hangzhou testnet.