Appendix – Protocol Semantics

1.1 Acronyms, Names and Token Colors

The acronyms, names and token colors of BlockVoke's CPN model are presented in Table 1. The first column specifies the module of the first occurrence of a certain acronym, name or token color, while the second column specifies its name. Next, the third column details the data type and structure. The last column provides a short description.

Table 1: Acronyms, names and token colors of BlockVoke's top-level CPN model.

Module	**Name**	**Type**	**Description**
	RSAPubKey	ColorSet (INT, INT)	Public Exponent and Modulus of an RSA Key
	RSAPrivKey	ColorSet (INT, INT)	Private Exponent and Public Modulus of RSA Key
	RSAKeyPair	ColorSet (RSAPubKey, RSAPrivKey)	RSA Keypair
Generate Certificate	Wallet_Addr	ColorSet (String)	A Wallet Address
	Wallet_PublicKey	String	A Wallet Public Key
	Wallet_PrivateKey	String	A Wallet Private Key
	Wallet_KeyPair	(Wallet_PublicKey, Wallet_PrivateKey)	A Wallet KeyPair
	Wallet_Previous Hash	String	Hash of the last transaction with output this Wallet address
	Wallet_Balance	INT	Current Balance Amount
Top Level	Wallet	(Wallet_Addr, Wallet_KeyPair, Wallet_Previous_Hash, Wallet_Balance)	A Wallet
Create Revocation Transactions	Wallet_List	Token Color List [Wallet]	A List of Wallets
	CO_CN	(String)	Common Name of a Certificate Owner
	CO_PublicKey	RSAPubKey	RSA Public Key of a CO
	CO_PrivateKey	RSAPrivateKey	RSA Private Key of a CO
	CA_CN	(String)	Common Name of a Certificate Authority
	CO_Key_ID	String	Unique Key ID according to RFC5280 Specification
Generate Certificate	CO_KeyPair	ColorSet (CO_CN, CO_PublicKey, CO_PrivateKey, CO_Key_ID)	CO's RSA Keypair

Table 1: Acronyms, names and token colors of BlockVoke's top-level CPN model.

Module	**Name**	**Type**	**Description**
	CA_PublicKey	RSAPubKey	RSA Public Key of a CA
	CA_PrivateKey	RSAPrivateKey	RSA Private Key of a CA
	CA_Key_ID	String	Unique Key ID according to RFC5280 Specification
Generate Certificate	CA_KeyPair	ColorSet (CA_CN, CA_PublicKey, CA_PrivateKey, CA_Key_ID)	CA's RSA Keypair
	Cert_Valid_From	ColorSet String	Date from which Certificate is valid
	Cert_Valid_To	ColorSet String	Date until which Certificate is valid
	Cert_Signed	ColorSet BOOL	Flag indicating if Certificate was signed
	Cert_Sig	ColorSet INT	Certificate Signature
	Cert_Fingerprint	INT	Certificate Fingerprint
Generate Certificate	Cert_Multisig_addr	String	Certificate Multisignature Address
Generate Certificate	Cert_DOI	String	Certificate Date of Issuance
Top Level	BlockVoke_Cert	ColorSet (CO_CN, CO_PublicKey, CO_Key_ID, CO_CN, CO_PublicKey, CO_Key_ID, Cert_Valid_From, Cert_Valid_To, Cert_Multisig_addr, Cert_Sig, Cert_Fingerprint, Cert_DOI	SSL Certificate with extra BlockVoke fields
Mark Certificate as Revoked	BlockVoke_Cert_List	List [BlockVoke_Cert]	List of Certificates
Top Level	CSR	ColorSet (BlockVoke_Cert, Wallet_Addr)	Representation of a Certificate Signing Request
	Funds	INT	Funds to be used for Revocation transactions

Table 1: Acronyms, names and token colors of BlockVoke's top-level CPN model.

Module	**Name**	**Type**	**Description**
	RFC5280RevocationCode	String	String Representation of a Revocation Code, as per RFC5280 Specification
	Fees	INT	Fees, twice the amount to be paid to each miner of each Revocation Transaction
	Is_CA	BOOL	Flag to specify if the CA is Revoking the Certificate
Top Level	RV	ColorSet (Cert_Fingerprint, Is_CA, Funds, Wallet_Addr, Fees, RFC5280_RevocationCode, Cert_Multisig_addr, Cert_DOI, CA_Key_ID)	ColorSet with information required for a Revocation
	ADDR	String	A Blockchain Address
	Hash	String	A Hash
	TX_Hash	Hash	A Transaction Hash
	TX_Prev_Hash	TX_Hash	Hash of Previous Transaction with Output to Input Address
	TX_Value	INT	Tokens spent in Transaction, subtracted from Total in Input Address
	TX_Output_Addr	ADDR	Output Address of the Transaction
	TX_Input_Addr	ADDR	Input Address of the Transaction
Add OP_RETURN Script	OP_RETURN	ColorSet (STRING, Cert_Fingerprint, Cert_DOI, RFC5280_RevocationCode, CA_Key_ID)	OP_RETURN Script of a Tx:Revoke Transaction

Table 1: Acronyms, names and token colors of BlockVoke's top-level CPN model.

Module	**Name**	**Type**	**Description**
Revoke Certificate	TX	ColorSet (TX_Hash, TX_Prev_Hash, TX_Value, Fees, TX_Output_Addr, OP_RETURN, TX_Input_Addr)	ColorSet representing a Transaction
Revoke Certificate	TX_PAIR	List [TX]	A pair of Revocation Transactions
Revoke Certificate	MEMPOOL	List [TX]	A list of Transactions, representing a Blockchain Mempool
Mine Revocation Transactions	NONCE	INT	Nonce Calculated for a new Block in a Blockchain
	BLOCK_HEADER	ColorSet (INT, NONCE)	ColorSet representing a Block Header
Revoke Certificate	TX_LIST	List [TX]	List of transactions in a Block
Revoke Certificate	BLOCK	ColorSet (BLOCK_HEADER, TX_LIST)	ColorSet representing a Block in a Blockchain
Mark Certificate as Revoked	bv_cert_list	Variable of color BlockVoke_Cert_List	Variable
Add OP_RETURN Script	opr	Variable of color OP_RETURN	Variable
Top Level	bv_cert	Variable of color BlockVoke_Cert	Variable
Generate Certificate	ca_addr	Variable of color Wallet_Addr	Variable
Create TX:Fund Transaction	tx	Variable of color TX	Variable
	tx1	Variable of color TX	Variable
	tx2	Variable of color TX	Variable
Create Revocation Transactions	wallet_list	Variable of color Wallet_List	Variable
Mark Certificate as Revoked	mempool_tx	Variable of colorTX	Variable

Table 1: Acronyms, names and token colors of BlockVoke's top-level CPN model.

Module	**Name**	**Type**	**Description**
Mark Certificate as Revoked	mined_tx	Variable of colorTX	Variable
Mine Revocation Transactions	block_number	Variable of color INT	Variable
Mine Revocation Transactions	tx_list	Variable of color TX_LIST	Variable
Mine Revocation Transactions	miner_mines	Variable of color BOOL	Variable
Generate Certificate	cert_doi	Variable of color Cert_DOI	Variable
Mine Revocation Transactions	nonce	Variable of color INT	Variable
Generate Certificate	wallet	Variable of color Wallet	Variable
Generate Certificate	co_wallet	Variable of color Wallet	Variable
Generate Certificate	ca_wallet	Variable of color Wallet	Variable
Generate Certificate	co_keypair	Variable of color CO_KeyPair	Variable
Generate Certificate	ca_keypair	Variable of colorCA_KeyPair	Variable
Generate Certificate	csr	Variable of color CSR	Variable
Generate Certificate	multisig_addr	Variable of color Cert_Multisig_addr	Variable
	tx_hash	Variable of color TX_Hash	Variable
Create Revocation Transactions	tx_fund	Variable of color TX	Variable
Create Revocation Transactions	tx_rev	Variable of color TX	Variable
Revoke Certificate	rv	Variable of color RV	Variable

Table 1: Acronyms, names and token colors of BlockVoke's top-level CPN model.

Module	**Name**	**Type**	**Description**
Create Revocation Transactions	tx_pair	Variable of color TX_PAIR	Variable
Create Revocation Transactions	tx_revoke	Variable of color TX	Variable
Add Unconfirmed Revocation Transactions to Mempool	mempool	Variable of color MEMPOOL	Variable
Mine Revocation Transactions	block	Variable of color BLOCK	Variable
Generate Certificate	valid	Variable of color BOOL	Variable

1.2 Functions

Table 2 shows the Functions defined in the CPN model, along with a brief description.

Table 2: Functions defined in CPN Model

Declaration	Description

`val hashtablesize = 2939 (* a prime*)`
`fun combine [] = 0`
`| combine (h :: t) = (ord h + 7 * combine t)`
`mod hashtablesize`
`fun hash s = combine (explode s);`
	Simplfied simulated hashing function

`fun poww(mu, n, e) = (if e = 1 then mu*n`
`else poww(mu*n, n, e-1));`
`fun pow(n, e) = poww(1, n, e);`
`fun modpoww(mul, a, b, n) =`
`(if b = 1 then (mul*a mod n)`
`else modpoww(mul*a mod n, a, b-1, n));`
`fun modpow(a, b, n) = modpoww(1, a, b, n);`
	Functions fascilitating simple exponentiation and modular exponentiation

`fun signRSA((n, d), H) = modpow(H, d, n);`
`fun verifyRSA((n,e),H,s) =`
`(H = modpow(s, e, n));`
	Functions fascilitating simplified RSA signing and signature verification

`fun addMultisigAndDate((cocn, copub, cokid,`
`cacn, capub, cakid, cvf, cvt, cma, cs,`
`cf, cdoa), ma, doa) =`
`(cocn, copub, cokid, cacn, capub,`
`cakid, cvf, cvt, ma, cs, cf, doa);`
	Function that adds multisig address and date of issuance to a BlockVoke_Cert

Table 2: Functions defined in CPN Model

Declaration	**Description**
`fun hashCert((cocn, (con, cod), cokid, cacn, (can, cad), cakid, cvf, cvt, cma, cs, cf, cdoa)) = hash( cocn ^ Int.toString(con) ^ Int.toString(cod) ^ cokid ^ cacn ^ Int.toString(can) ^ Int.toString(cad) ^ cakid ^ cvf ^ cvt ^ cma ^ cdoa);`	Function that returns hash of a BlockVoke_Cert
`fun signCert((cocn, (con, cod), cokid, cacn, (can, cad), cakid, cvf, cvt, cma, cs, cf, cdoa), (n, d)) = (cocn, (con, cod), cokid, cacn, (can, cad), cakid, cvf, cvt, cma, signRSA((n, d), hashCert(cocn, (con, cod), cokid, cacn, (can, cad), cakid, cvf, cvt, cma, cs, cf, cdoa)), 0, cdoa);`	Function that returns a signed BlockVoke_Cert using an RSAPrivKey

Table 2: Functions defined in CPN Model

Declaration	Description
`fun computeCertF((cocn, (con, cod), cokid, cacn, (can, cad), cakid, cvf, cvt, cma, cs, cf, cdoa)) =hash(cocn ^ Int.toString(con) ^ Int.toString(cod) ^ cokid ^ cacn ^ Int.toString(can) ^ Int.toString(cad) ^ cakid ^ cvf ^ cvt ^ cma ^ Int.toString(cs) ^ cdoa);`	Function that computes the Fingerprint of a signed BlockVoke_Cert
`fun setCertF((cocn, (con, cod), cokid, cacn, (can, cad), cakid, cvf, cvt, cma, cs, cf, cdoa)) = (cocn, (con, cod), cokid, cacn, (can, cad), cakid, cvf, cvt, cma, cs, computeCertF((cocn, (con, cod), cokid, cacn, (can, cad), cakid, cvf, cvt, cma, cs, cf, cdoa)), cdoa);`	Function that sets the fingerprint of a BlockVoke_Cert
`fun verifyCert(cf, h, (n, e)) = verifyRSA((n, e), h, cf);`	Function that verifies the signature of a BlockVoke_Cert using an RSAPubKey
`fun TX_setPrevTX((th, tph, tv, f, txoa, opr, txia), (wa, wkp, wph, wb)) = (th, wph, tv, f, txoa, opr, txia);`	Function that sets the TX_Prev_Hash of a TX, using a Wallet as input

Table 2: Functions defined in CPN Model

Declaration	Description
`fun TX_setCredits((th, tph, tv, f, txoa, opr, txia), value, fees) = (th, tph, value, fees, txoa, opr, txia);`	Function that sets the TX_Value and Fees being spent to a TX
`fun TX_setInputAddr((th, tph, tv, f, txoa, opr, txia), (wa, wkp, wph, wb)) = (th, tph, tv, f, txoa, opr, wa);`	Function that sets the TX_Input_Addr of a TX, using a Wallet as input
`fun TX_setOutputAddr((th, tph, tv, f, txoa, opr, txia), oa) = (th, tph, tv, f, oa, opr, txia);`	Function that sets the TX_Output_Addr of a TX
`fun TX_setOPR((th, tph, tv, f, txoa, txopr, txia), opr) = (th, tph, tv, f, txoa, opr, txia);`	Function that sets the OP_RETURN value of a TX
`fun OPR_setBVI((oprbvi, oprcf, oprcdoa, oprrfc, oprcki), bvi) = (bvi, oprcf, oprcdoa, oprrfc, oprcki);`	Function that sets the BlockVoke Identifier in an OP_RETURN script
`fun OPR_setCertF((oprbvi, oprcf, oprcdoa, oprrfc, oprcki), (cf, ic, fu, wa, ff, rfc, cma, cdoa, cki)) = (oprbvi, cf, oprcdoa, oprrfc, oprcki);`	Function that sets the Cert_Fingerprint of an OP_RETURN Script

Table 2: Functions defined in CPN Model

Declaration	Description
`fun OPR_setCertDOA((oprbvi, oprcf, oprcdoa, oprrfc, oprcki), (cf, ic, fu, wa, ff, rfc, cma, cdoa, cki)) = (oprbvi, oprcf, cdoa, oprrfc, oprcki)`	
`fun OPR_setRFC((oprbvi, oprcf, oprcdoa, oprrfc, oprcki), (cf, ic, fu, wa, ff, rfc, cma, cdoa, cki)) = (oprbvi, oprcf, oprcdoa, rfc, oprcki);`	Function that sets the RFC5280_RevocationCode of an OP_RETURN script
`fun OPR_setCAKID((oprbvi, oprcf, oprcdoa, oprrfc, oprcki), (cf, ic, fu, wa, ff, rfc, cma, cdoa, cki)) = (oprbvi, oprcf, oprcdoa, rfc, cki);`	Function that sets the CA_Key_ID of an OP_RETURN script
`fun hashTX((txh, txph, txv, f, txoa, (bvi, cf, cdoa, rfc,cki), txia)) = "0x" ^ Int.toString(hash( txh^ txph^ Int.toString(txv)^ Int.toString(f)^ txoa^ bvi^ Int.toString(cf)^ cdoa^ rfc^ cki^ txia));`	Function that computes TX_Hash of a TX

Table 2: Functions defined in CPN Model

Declaration	Description
`fun hashedTX((txh, txph, txv, f, txoa, (bvi, cf, cdoa, rfc, cki), txia)) = (hashTX(txh, txph, txv, f, txoa, (bvi, cf, cdoa, rfc, cki), txia), txph, txv, f, txoa, (bvi, cf, cdoa, rfc,cki), txia);`	Function that sets the TX_Hash of a TX
`fun updateWallet((wallet_addr, wallet_keypair, wallet_prev_hash, wallet_balance), new_tx_hash, sub_amount) = (wallet_addr, wallet_keypair, new_tx_hash, wallet_balance- sub_amount);`	Function that updates a Wallet_Balance by subtracting a given amount and Wallet_Previous_Hash of a Wallet
`fun mineBlock(mempool, number, nonce) = ((number, nonce), mempool)`	Function that creates a BLOCK using a TX_List(Mempool), a number and a NONCE
`fun checkmultisig(walletaddr, maddr) = if (maddr = "0xmultisig1" andalso walletaddr = "0x3") then true else (if maddr = "0xmultisig2" andalso walletaddr = "0x4" then true else ((if maddr = "0xmultisig3" andalso walletaddr = "0x5" then true else ((if maddr = "0xmultisig4" andalso walletaddr = "0x6" then true else false)))));`	Function that returns a multisig address, specific to a wallet