DESCRITIVO TÉCNICO – TOKEN XGO (ConexaGo)

Versão de Teste – MVP (Fase Beta)

v1.0 – Julho de 2025

---

1. Identificação da Obra

- **Título da Obra:** Whitepaper Técnico – ConexaGo Token (XGO)
- **Data da Publicação:** 29 de julho de 2025
- **Registro público:**
  Publicly published on GitHub at:
  https://github.com/emilioconexa/ConexaGo-Desenvolvimento-Ltda/blob/main/Whitepaper%20Consolidado%20v1.0%20072025.pdf
- 2b928f2117d06a4e0c907a76ba184e538b1dcc9c219bec26527dba2363b839d8
- **Responsável técnico e autor intelectual:**
  Emilio Cesar Cleris Rossi
- **Empresa titular dos direitos:**
  ConexaGo Desenvolvimento Ltda.
- **CNPJ:** 41.401.810/0001-37

---

2. Informações Técnicas do Token

Campo	Valor
Nome do Token	ConexaGo Token
Símbolo (Ticker)	XGO
Padrão	ERC-20 (com funções restritivas)
Casas decimais	18
Supply Total	1.000.000.000 XGO
Rede de Deploy	Polygon Testnet (Amoy)

Campo	Valor
Endereço do Contrato	0xc50fc5DfA7C6D4A020bE08E17E4Ce3B4fd27AAfa
Endereço do Owner	0x5FEf06C8834e1F91192d6985BF6E4592F6C28b0E
Data do Deploy	29/07/2025
Visualizador Blockchain	Amoy Polygonscan

---

3. Política de Distribuição Inicial

Categoria	Percentual	Quantidade (XGO)	Observações
Uso Interno	90%	900.000.000	Exclusivo para recompensas e resgates
Reserva Estratégica	10%	100.000.000	Marketing, expansão e liquidez futura
Total Distribuído	**100%**	**1.000.000.000**	Total em circulação no contrato

---

4. Funcionalidades do Contrato Inteligente

- **transfer(address to, uint256 value):**
  Transfere tokens entre carteiras da whitelist.
- **burn(uint256 value):**
  Permite que qualquer carteira whitelist queime seus próprios tokens.
- **addToWhitelist(address _addr):**
  Adiciona endereço autorizado para transações.
- **removeFromWhitelist(address _addr):**
  Remove autorização de endereço.
- **getMyBalance():**
  Retorna o saldo da carteira chamadora.
- **balanceOf(address):**
  Consulta saldo de qualquer carteira.

- **totalSupply():**
  Consulta total de tokens em circulação.
- **internalUseWallet():**
  Endereço que recebeu os 90%.
- **strategicReserveWallet():**
  Endereço que recebeu os 10%.
- **owner():**
  Mostra o proprietário atual do contrato.

---

5. Contrato-Fonte – Trecho Inicial (XGOToken.sol)

solidity
CopiarEditar

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract XGOToken {
    string public name = "ConexaGo Token";
    string public symbol = "XGO";
    uint8 public decimals = 18;
    uint256 public totalSupply;

    address public owner;
    address public internalUseWallet;
    address public strategicReserveWallet;

    mapping(address => uint256) public balanceOf;
    mapping(address => bool) public whitelist;

    event Transfer(address indexed from, address indexed to, uint256 value);
    event Burn(address indexed from, uint256 value);

    modifier onlyOwner() {
        require(msg.sender == owner, "Apenas o owner");
        _;
```

```
}

modifier onlyWhitelisted(address from, address to) {
    require(whitelist[from] && whitelist[to], "Transferencias bloqueadas");
    _;
}

constructor(address _internalUseWallet, address _strategicReserveWallet) {
    require(_internalUseWallet != address(0), "Endereco interno invalido");
    require(_strategicReserveWallet != address(0), "Endereco estrategico invalido");

    owner = msg.sender;
    internalUseWallet = _internalUseWallet;
    strategicReserveWallet = _strategicReserveWallet;

    uint256 _total = 1_000_000_000 * 10**uint256(decimals);
    totalSupply = _total;

    uint256 internalAmount = (_total * 90) / 100;
    uint256 strategicAmount = _total - internalAmount;

    balanceOf[_internalUseWallet] = internalAmount;
    balanceOf[_strategicReserveWallet] = strategicAmount;

    whitelist[_internalUseWallet] = true;
    whitelist[_strategicReserveWallet] = true;
    whitelist[msg.sender] = true;

    emit Transfer(address(0), _internalUseWallet, internalAmount);
    emit Transfer(address(0), _strategicReserveWallet, strategicAmount);
}

function transfer(address to, uint256 value) public onlyWhitelisted(msg.sender, to) returns (bool) {
    require(balanceOf[msg.sender] >= value, "Saldo insuficiente");
    require(to != address(0), "Endereco invalido");

    balanceOf[msg.sender] -= value;
    balanceOf[to] += value;
```

```
        emit Transfer(msg.sender, to, value);
        return true;
    }

    function burn(uint256 value) public returns (bool) {
        require(balanceOf[msg.sender] >= value, "Saldo insuficiente para queima");

        balanceOf[msg.sender] -= value;
        totalSupply -= value;

        emit Burn(msg.sender, value);
        return true;
    }

    function addToWhitelist(address _addr) public onlyOwner {
        whitelist[_addr] = true;
    }

    function removeFromWhitelist(address _addr) public onlyOwner {
        whitelist[_addr] = false;
    }

    function getMyBalance() external view returns (uint256) {
        return balanceOf[msg.sender];
    }
}
```

Notas Finais

- O contrato está operando em ambiente **Testnet** para validação do MVP.
- Todos os dados foram publicados em ambiente público e registrado com hash.
- Futuras versões serão versionadas com novos registros e nova hash.

Nota de Registro Público – Descritivo Técnico

Este Descritivo Técnico foi publicado publicamente em 29 de julho de 2025 com o objetivo de comprovação de autoria, integridade e anterioridade do projeto **ConexaGo Token (XGO)**.

REGISTRO DE INTEGRIDADE – HASH SHA-256

Para garantir a **integridade** e a **originalidade** deste whitepaper, foi gerado um **hash SHA-256** do arquivo PDF final, publicado publicamente em **30 de julho de 2025**.

Hash SHA-256: []

Publicado publicamente no GitHub em: []

Qualquer alteração no conteúdo invalida esta assinatura.
Isso assegura **rastreabilidade**, **proteção autoral** e **transparência**.

Fim do documento.