MaxProcess – System Summary

System: MaxProcess

Document Type: Strategic Summary

Version: 1.0

Responsible: TBYD Architectural Team / Process Capsule Group

License Model: TBYD License v2.2 / Audit Addendum A

Standards Reference: ISO/IEC 15408, AIM v1.0, TBYD Capsule Protocol v2.1

Applicability: MaxOneOpen v4.1+

# 1. Purpose and Scope

MaxProcess is the sovereign process execution subsystem of MaxOneOpen. It introduces a new form of runtime-validated, capsule-bound workflow execution that is audit-traceable, treaty-aware, and architecturally detached from conventional process engines. MaxProcess does not execute code – it governs the structural, auditable flow of authority and process decisions. Every process step is formalized as a signed, runtime-verifiable capsule with full rollback and refusal logic, governed by role, policy, and treaty contexts.  
  
The system provides a structural alternative to traditional business process management (BPM), workflow suites, or runtime schedulers. It prioritizes procedural auditability, actor accountability, and formal treaty control in digitally sovereign environments. MaxProcess enables OSS, institutions, or cross-party federations to define and enforce process logic that is fully observable, revocable, and policy-conform — without relying on hidden execution paths or commercial orchestration platforms.

# 2. Functional Architecture

MaxProcess defines all executable logic as cryptographically signed ProcessCapsules. Each capsule represents a structured, traceable execution unit that includes:  
- a defined process step or decision point  
- a responsible role (via MaxWorkRoles binding)  
- a trigger or condition (time, input, event, prior capsule state)  
- optional repeat logic (loop, interval, cascade)  
- fallback or refusal path (if rejected or untriggered)  
- linkage to audit and governance channels  
  
The execution of a ProcessCapsule does not imply the execution of code or automation in a classical sense. Instead, it formalizes procedural responsibility and outcome structure. It determines \*who must act\*, \*what must be confirmed or escalated\*, and \*how each step is embedded in an audit-verifiable trail\*.  
  
Capsules form directed capsule chains with forward and backward trace references. State is never mutable but rather inherited by confirmation or rule-based continuation. All process chains are exportable, importable, and signable, including their context (e.g., treaty, environment, execution class). This design ensures fork resistance, external observability, and runtime override through governance-linked actors.

# 3. Role Binding and Runtime Control

MaxProcess operates under strict runtime role enforcement. Every ProcessCapsule defines a required role, which must be resolved through MaxWorkRoles at the time of execution. This ensures that only authorized, currently valid, and treaty-conform actors may initiate or complete any procedural step.  
  
Capsule execution is blocked unless the role is:  
- active and signed (with valid capsule lineage)  
- permitted in the current tenant or treaty scope  
- not shadowed or revoked (by governance or audit action)  
  
Roles may be refused (explicitly) or skipped (if fallback logic exists), triggering alternate flows or capsule freezing. Execution traces are logged in MaxAudit, including:  
- operator identity reference  
- step confirmation or refusal  
- timestamp and scope context  
- associated governance conditions (if overridden or escalated)  
  
Runtime control allows systems to dynamically adapt process behavior based on live role state, audit history, treaty policy, or process priority class. No capsule may execute outside these constraints.