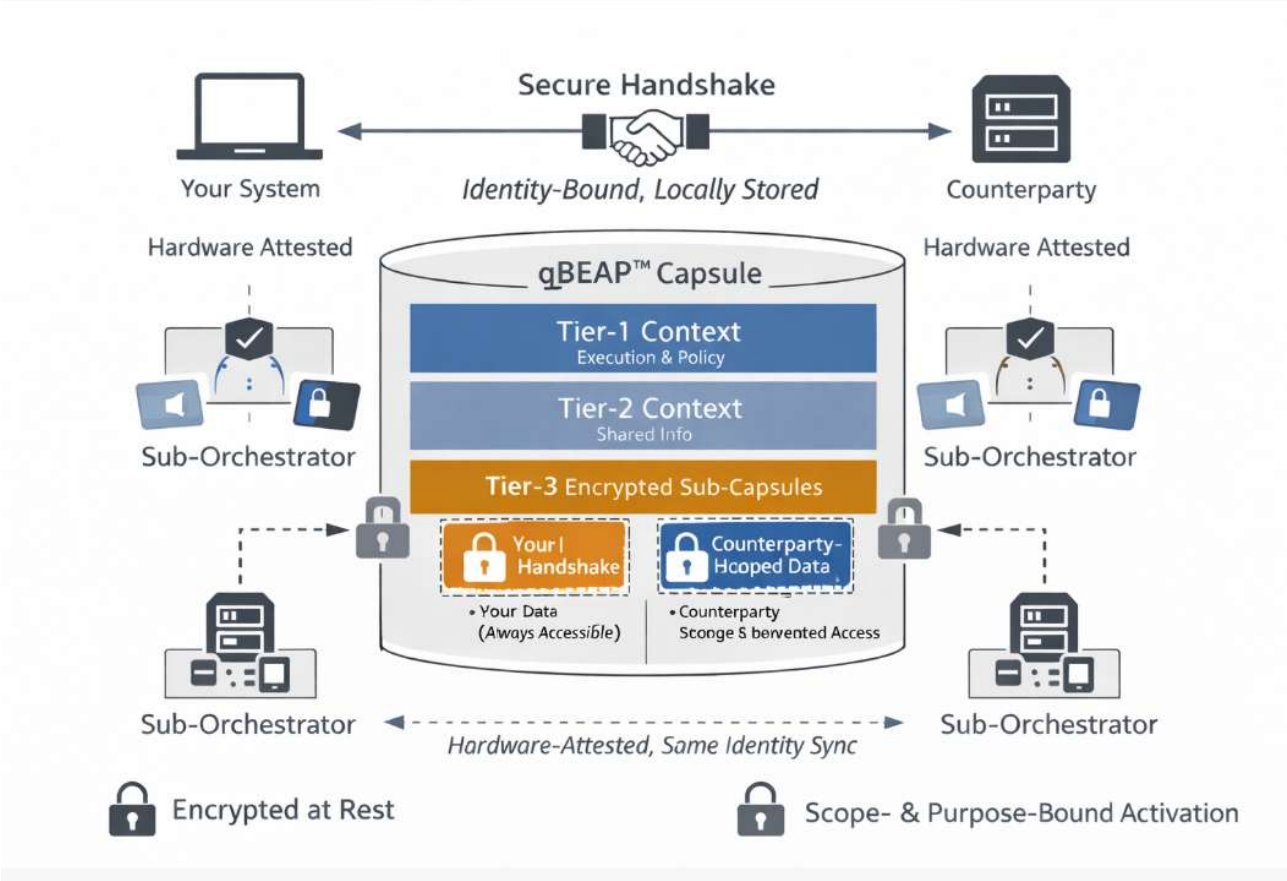

Annex A1 (Normative)

Expectation-Driven Handshake Synchronization, Attested Execution Environments, and Purpose-Bound Data Activation in BEAP™

A1.1 Status and Scope

This annex is normative.
It defines mandatory handshake semantics, storage rules, access boundaries, and synchronization constraints for BEAP™-compliant implementations that support expectation-driven handshakes and handshake-scoped data.

This annex does not modify or extend:

- wire formats,
- capsule serialization formats,
- cryptographic primitives,
- transport mechanisms, or
- PoAE™ execution semantics

defined in the BEAP™, qBEAP™, and PoAE™ specifications.

This annex defines behavioral, architectural, and security requirements that MUST be honored by conforming implementations.

A1.2 Applicability and Environment Constraints

The handshake model defined in this annex SHALL impose hardware-attestation requirements only on environments that request, activate, process, or receive sensitive effects under a handshake-scoped cooperation state.

An implementation MUST NOT enable processing, activation, or access to handshake-scoped sensitive data unless:

- the requesting or processing environment is hardware-attested, and
- the attestation is bound to a verifiable identity recognized by the BEAP™ trust model.

Hardware attestation:

- SHALL be mandatory for environments that request, activate, or process:
    - PII,
    - sensitive data,
    - financial or other irreversible effects.
- SHALL NOT be mandatory for environments that:
    - initiate first contact,
    - negotiate expectations,
    - express consent,
    - participate in handshake establishment without processing sensitive data.

Non-attested environments MAY participate in BEAP™ interactions and MAY be parties to a handshake but SHALL NOT request, activate, process, or receive handshake-scoped sensitive data.

A1.3 Handshake as an Operational Boundary

In BEAP™, a handshake SHALL represent a bounded, identity-anchored cooperation state shared between two parties.

A handshake state SHALL be:

- identity-bound,
- explicitly consented,
- cryptographically verifiable,
- revocable,
- and confined to attestation-enforced execution boundaries for sensitive operations.

The handshake SHALL function as a temporary operational contract governing:

- which data classes MAY exist within exchanged capsules,
- which expectations apply,

- which encrypted data regions MAY be accessed,
- and under which declared purposes activation MAY occur.

All shared context SHALL be strictly scoped to the lifetime and declared scope of the handshake.

A1.4 Local Handshake Persistence

Each participating party SHALL store the handshake state locally.

Local handshake persistence:

- SHALL be encrypted at rest,
- SHALL be cryptographically bound to the handshake identifier,
- SHALL be bound to the local execution environment and identity,
- SHALL NOT constitute a global session state,
- SHALL NOT require continuous online availability of the counterparty.

For handshake-scoped sensitive data, local persistence and any subsequent activation SHALL require a hardware-attested execution environment.

Local persistence SHALL NOT relax capsule-local determinism or PoAE™ policy enforcement.

A1.5 Expectation-Driven Synchronization

A1.5.1 Expectations as Declarative Signals

An expectation SHALL be a declarative statement describing information that MAY be required for a handshake to transition into an active operational state.

An expectation:

- SHALL NOT mandate immediate disclosure,
- SHALL NOT imply processing,
- SHALL define requirements, not actions.

An expectation MAY specify:

- attribute identifiers,
- data class (e.g. PII, business-critical, informational),
- purpose identifiers,
- requirement level,
- additional constraints (e.g. jurisdiction, format, validation state).

A1.5.2 Iterative Capsule Synchronization

Handshake completion SHALL occur through iterative capsule exchange.

Capsules exchanged during synchronization MAY contain:

- capsule-local deterministic execution context,
- shared informational references,
- encrypted handshake-scoped data regions.

Sensitive data SHALL NOT be required to be processed solely to satisfy expectation negotiation.

A1.6 qBEAP™ Capsule Structure and Encrypted Regions

A qBEAP™ capsule used under an active handshake MAY contain multiple encrypted regions, each with distinct access semantics.

A1.6.1 Capsule Regions

A capsule MAY conceptually contain:

Region 1 — Tier-1 Context
Capsule-local deterministic execution context and policy references.
This region SHALL be the sole authority for execution and automation decisions.

Region 2 — Tier-2 Context
Shared informational and non-sensitive material, including augmented overlay annotations and support context.

Region 3 — Handshake-Scoped Encrypted Sub-Capsules
One or more encrypted sub-capsules containing handshake-scoped data such as PII, sensitive data, or augmented overlay support for sensitive operations.

A1.6.2 Handshake-Scoped Encrypted Sub-Capsules

Handshake-scoped data SHALL be stored exclusively within encrypted qBEAP™ sub-capsules.

Each such sub-capsule:

- SHALL be encrypted at rest and in transit,
- SHALL be cryptographically bound to the handshake identifier,
- SHALL be isolated from execution context,
- SHALL NOT influence execution logic,
- SHALL be inaccessible without explicit activation.

Multiple encrypted sub-capsules MAY coexist within a single qBEAP™ capsule.

A1.7 Access Semantics and Purpose-Bound Activation

Processing or access to handshake-scoped sensitive data SHALL be permitted only when all of the following conditions are met:

- a matching expectation exists,
- a declared purpose identifier applies,
- policy conditions are satisfied,

- valid consent is present,
- the requesting or processing environment is hardware-attested.

Requesting or activating access to PII, sensitive data, or irreversible effects without hardware attestation SHALL NOT be permitted, regardless of consent or handshake existence.

Each party:

- SHALL be able to decrypt and access its own contributed handshake-scoped data at any time,
- SHALL NOT gain unrestricted access to counterparty-contributed data.

Access to counterparty-provided handshake data:

- SHALL be scope-bound,
- SHALL be purpose-bound,
- SHALL be denied by default.

Transport or storage of encrypted handshake-scoped data SHALL NOT constitute processing.

A1.8 Sub-Orchestrator Synchronization

Handshake-scoped data MAY be synchronized between sub-orchestrators operating under the same identity, provided that:

- each sub-orchestrator that may activate or process sensitive data executes in a hardware-attested environment,
- attestation confirms equivalence of trust level,
- the identity binding is verifiable and unchanged.

Synchronization MAY include:

- an empty qBEAP™ capsule,
- the handshake identifier,
- encrypted handshake-scoped sub-capsules.

Sub-orchestrators that are not hardware-attested MAY receive handshake metadata but SHALL NOT receive, activate, or process handshake-scoped sensitive data.

Synchronization SHALL NOT relax access controls or activation rules defined in this annex.

A1.9 Permitted Data Introduction Paths

BEAP™ SHALL permit two normative mechanisms for introducing sensitive data into a handshake-scoped cooperation state:

1. Expectation fulfillment via explicit data request
2. Handshake-anchored encrypted data presence

In both cases, data SHALL remain inactive until purpose-bound activation occurs.

A1.10 Security, Privacy, and Determinism Properties

The model defined in this annex:

- restricts processing to explicitly declared purposes,
- prevents implicit or ambient data access,
- preserves auditability and traceability,
- enforces role- and operation-bound attestation requirements,
- maintains strict determinism.

Handshake revocation SHALL immediately affect future access and synchronization without altering historical integrity.

A1.12 First Contact and Handshake Eligibility

A BEAP™ interaction MAY begin without an existing handshake.

First contact MAY occur via:

- public WR Codes,
- websites or applications,
- BEAP™ Packages containing pBEAP™ capsules,
- other BEAP™-compatible interaction mechanisms.

The absence of a handshake SHALL NOT prevent:

- expectation declaration,
- message exchange,
- automation execution,
- data requests,
- consent negotiation,
- or handshake establishment.

qBEAP™ capsules SHALL NOT be used for first contact. qBEAP™ capsules SHALL be permitted only for enterprise handshakes in which both parties execute in hardware-attested environments.

A1.13 Handshake Creation and Promotion

A handshake MAY be created as a result of an interaction that includes:

- a declared expectation,
- an explicit data request,

- and valid user consent.

When these conditions are met, an implementation MAY promote the interaction into a handshake-scoped cooperation state.

Handshake creation:

- SHALL require explicit, informed consent,
- SHALL be identity-bound,
- SHALL be cryptographically verifiable,
- SHALL be locally persisted,
- SHALL be revocable at any time.

No prior authentication or handshake SHALL be required for handshake creation itself.

Handshake promotion SHALL NOT:

- imply immediate processing of sensitive data,
- bypass purpose or policy constraints,
- activate sensitive data without satisfying the requirements defined in this annex.

A1.14 Handshake Scope Anchoring and Reuse

Each handshake SHALL be anchored to a declared cooperation scope.

A handshake scope SHALL define:

- the cooperating counterparty identity,
- the permitted interaction domain or service context,
- the allowed purpose identifiers,
- the permitted data classes.

A handshake SHALL NOT be implicitly bound to a specific WR Code.

Multiple interaction entry points, including multiple public WR Codes and BEAP™ Packages, MAY reference the same handshake, provided their declared scope requirements are compatible with the handshake scope.

Scope compatibility SHALL be evaluated deterministically at execution time.

A1.15 Capsule Formats, Automation, and Attestation Requirements

BEAP™ defines multiple capsule formats with distinct security and deployment properties.

A pBEAP™ capsule:

- SHALL be unencrypted,
- MAY carry messages, expectations, and automation instructions,

- SHALL be used for first contact, public WR Code interactions, and general messaging,
- SHALL NOT require hardware attestation on both parties.

For public publishers of WR Codes, the publishing environment:

- SHALL be hardware-attested,
- SHALL be identity-bound within the BEAP™ trust model,
- SHALL NOT require the scanning or consuming party to be hardware-attested at first contact; however, additional use cases MAY require hardware attestation of the scanning or consuming party after handshake establishment.

A qBEAP™ capsule:

- SHALL be encrypted,
- MAY carry messages and automation instructions,
- SHALL be used exclusively for enterprise handshakes,
- SHALL require both parties to execute in hardware-attested environments.

Automation capability is not limited to qBEAP™ and MAY be executed within pBEAP™ capsules, subject to policy and trust constraints.

A1.16 Encrypted Handshake-Scoped Sensitive Data Containers

In addition to capsule formats, an encrypted handshake-scoped sensitive data container MAY be used as an edge case for sensitive data handling.

Such encrypted handshake context:

- SHALL contain sensitive data only,
- SHALL NOT contain automation templates or execution logic,
- SHALL NOT constitute a qBEAP™ capsule,
- MAY be embedded within a BEAP™ Package containing a pBEAP™ capsule,
- SHALL require hardware attestation only for the requesting or processing environment,
- SHALL be encrypted using the same cryptographic construction, key management model, and post-quantum–ready primitives as defined for qBEAP™ capsules.

The existence of such encrypted containers SHALL NOT alter the classification of the enclosing capsule format.

A1.17 Sensitive Data Introduction During Handshake Establishment

Sensitive data MAY be introduced during or after handshake establishment only when:

- a matching expectation exists,
- a declared purpose identifier applies,

- explicit consent is granted,
- the requesting or processing environment satisfies the hardware-attestation requirements defined in this annex.

Sensitive data introduced under a handshake:

- SHALL be stored exclusively within encrypted handshake-scoped containers,
- SHALL remain inactive until purpose-bound activation occurs,
- SHALL NOT influence execution logic.

Requesting or activating access to PII or sensitive data without hardware attestation SHALL NOT be permitted, regardless of consent or handshake existence.

A1.18 Public WR Codes and Handshake Interaction

Public WR Codes:

- SHALL remain static and unencrypted,
- SHALL resolve to BEAP™ Packages containing pBEAP™ capsules,
- SHALL NOT contain sensitive data,
- SHALL NOT embed handshake identifiers or user-specific state.

Personalized behavior resulting from scanning a public WR Code:

- SHALL derive exclusively from an existing handshake,
- SHALL be enforced through scope and purpose evaluation,
- SHALL NOT alter the semantic meaning of the WR Code.

A1.19 Handshake Revocation Effects

A handshake MAY be revoked at any time by either party.

Upon revocation:

- future access to encrypted handshake-scoped containers SHALL be denied,
- synchronization of handshake-scoped sensitive data SHALL cease,
- activation of encrypted handshake-scoped data SHALL be prevented.

Revocation SHALL NOT alter historical BEAP™ Package or capsule integrity or auditability.

A1.20 Handshake Establishment via Public WR Code®

Public WR Code® instances SHALL function as static, unencrypted, replay-safe entry points into BEAP™ interactions.

They SHALL NOT embed handshake state, encrypted material, user-specific information, dynamic identifiers, or mutable execution logic.

Public WR Code® interactions SHALL require network access for:

- WR Code® resolution,
- retrieval of referenced templates and BEAP™ Packages,
- integrity verification of retrieved artefacts,
- verification of externally anchored integrity commitments as defined by the WR Code® protocol.

Resolution of a public WR Code® SHALL always result in a pBEAP™-based interaction, independent of whether a handshake exists.

A1.20.1 Resolution and Initial Execution Context

Upon resolution of a public WR Code®, the orchestrator SHALL:

- resolve the publisher identity associated with the WR Code®,
- retrieve the referenced BEAP™ Package containing a pBEAP™ capsule,
- perform integrity verification of retrieved artefacts against externally anchored commitments,
- establish a deterministic execution context.

At this stage:

- no handshake SHALL be assumed to exist,
- no encrypted handshake-scoped context SHALL be active,
- no authentication or hardware attestation SHALL be implied for the consuming party.

The publishing environment of public WR Code® instances SHALL be hardware-attested and identity-bound within the BEAP™ trust model.

A1.20.2 Handshake Evaluation and Verifiable Continuity

Once the required artefacts and verification material are available, the orchestrator SHALL evaluate whether a locally stored handshake exists that:

- is identity-bound to the resolved publisher,
- declares a cooperation scope compatible with the current interaction.

This evaluation:

- SHALL be performed within the orchestrator,
- SHALL be deterministic with respect to the retrieved artefacts and local handshake state,

- MAY involve verification of externally anchored commitments to ensure that the handshake state and scope have not been altered retroactively.

Handshake evaluation and establishment MAY rely on external verification mechanisms but SHALL NOT delegate execution authority, policy decisions, or access control to external systems.

If no compatible handshake exists, execution SHALL proceed in generic, non-personalized mode.

A1.20.3 Expectation Declaration and Handshake Eligibility

A pBEAP™ capsule resolved from a public WR Code® MAY declare expectations indicating:

- intended cooperation scope,
- data classes that MAY be requested,
- purposes that would apply if cooperation were elevated.

Such expectations:

- SHALL remain declarative,
- SHALL NOT imply processing,
- SHALL NOT activate sensitive data,
- SHALL NOT require hardware attestation.

Expectations MAY designate the interaction as handshake-eligible but SHALL NOT, by themselves, establish a handshake.

A1.20.4 Handshake Establishment as a Verifiable State Transition

A handshake SHALL be established only when:

- explicit user consent is provided,
- the cooperating publisher identity is known,
- the declared scope and purposes are accepted.

Handshake establishment:

- SHALL be implemented as a local, deterministic state transition within the orchestrator,
- SHALL create an identity-bound and scope-bound handshake record,
- SHALL NOT require modification of the WR Code®,
- SHALL NOT require modification of the capsule format,
- SHALL NOT require mutual hardware attestation.

To prevent repudiation, silent alteration, or post-hoc denial, the establishment of a handshake and its declared scope SHALL be cryptographically committed and externally anchored in a privacy-preserving manner.

Such anchoring:

- SHALL record only cryptographic commitments (e.g. salted hashes or aggregated roots),
- SHALL NOT expose identities, scopes, purposes, or user-specific metadata,
- SHALL serve as non-authoritative proof of existence and integrity.

The handshake SHALL become the continuity anchor for subsequent interactions with the same publisher and compatible scope.

A1.20.5 Post-Handshake Capability Escalation

After handshake establishment, additional capabilities MAY become available, including:

- personalization of subsequent pBEAP™ interactions,
- storage and retrieval of encrypted handshake-scoped sensitive data containers,
- escalation of hardware attestation requirements for specific purposes or operations.

Such escalation:

- SHALL be strictly scope- and purpose-bound,
- SHALL require hardware attestation only for environments requesting or processing sensitive or irreversible actions,
- SHALL be verifiably attributable to the established handshake,
- SHALL NOT retroactively affect prior interactions.

A1.20.6 Integrity Anchoring and Non-Repudiation Guarantees

Externally anchored integrity commitments SHALL be used to ensure that:

- publishers cannot deny or retroactively alter offered templates, scopes, or handshake terms,
- consumers can prove the origin, scope, and integrity of interactions they accepted,
- service providers and integrators can demonstrate non-manipulation of execution logic.

Such anchoring mechanisms:

- SHALL be used solely for verification and auditability,
- SHALL NOT become authoritative for execution, authorization, or policy enforcement,
- SHALL preserve unlinkability between independent interactions,
- SHALL prevent both unilateral repudiation and fabricated claims.

This anchoring model is intended to eliminate ambiguity, abuse, and post-hoc denial across all parties, thereby preventing unbounded or adversarial behavior in public WR Code®–based interactions.

A1.11 Conclusion (Final, Corrected)

This annex defines a deliberate architectural shift for **BEAP™ communications and automation** away from implicit trust, transient sessions, and unverifiable exchanges toward a **handshake-anchored cooperation model** that is explicit, auditable, and resistant to repudiation by any party.

The principles defined herein apply uniformly across **all BEAP™ interaction mechanisms**, including but not limited to public **WR Code®**–initiated interactions, application-based exchanges, service-to-service communication, and automated workflows.
WR Code® is treated as a prominent and security-critical entry mechanism, but not as a limitation of the handshake or context model itself.

The mechanisms specified in this annex intentionally separate:

- **interaction initiation** (which may occur through multiple BEAP™-compatible channels),
- **cooperation continuity** (established through identity- and scope-bound handshakes),
- **capability escalation** (controlled by purpose-, role-, and attestation-bound activation),
- and **accountability** (provided through cryptographic integrity anchoring).

By anchoring context, expectations, and sensitive cooperation state to explicit handshakes rather than to transport artefacts, session constructs, or user interface entry points, the model enables persistent, verifiable cooperation without embedding mutable or user-specific state into messages, capsules, or automation logic.

Beyond its security properties, this model is designed to **reduce friction in everyday operational routines**.
Data required to satisfy handshake expectations is maintained locally within the **WRVault™** and does not need to be re-entered or reconfigured for each handshake.
Handshake establishment can therefore draw from an existing, locally controlled data base, while access to that data remains strictly scope- and purpose-bound.

Knowledge bases, references, and previously consented data can be queried efficiently once a handshake exists, without repeated disclosure or negotiation.
Authentication and authorization can occur quickly and deterministically, as all required information is already locally available, yet remains inaccessible outside the declared scope and purpose.

A central design goal of this annex is to eliminate ambiguity and post-hoc denial in distributed and cross-organizational environments.
Both parties to a BEAP™ interaction are protected against false claims, silent modification, and retroactive reinterpretation of scope, consent, or permitted actions, while integrators and platform operators retain a defensible, non-authoritative audit trail.

The use of externally anchored cryptographic commitments is intentional and normative.
Such anchoring provides **proof of existence and integrity**, not authorization or control, and serves as a neutral witness for handshake-scoped cooperation states that would otherwise be unverifiable in adversarial, regulated, or high-risk contexts.

This annex defines the **target architecture** of the WR Desk™ and BEAP™ ecosystem.
It is expected that early implementations of WR Desk™ may not yet implement all primitives, enforcement points, or anchoring mechanisms described herein.
Such partial implementations do not weaken or reinterpret this specification; rather, they reflect a staged evolution toward the fully verifiable, non-repudiable cooperation model defined by this annex.

Conforming implementations are expected to evolve toward full compliance over time, without diluting the guarantees, boundaries, or intent established herein.

WRVault™ (Terminology Clarification — Single Sentence, Normative)

WRVault™ denotes a locally controlled, encrypted, multi-layered secure vault bound to the executing environment and identity, combining password management, PII storage, and sensitive data protection, in which all data **SHALL** be compartmentalized, scope- and purpose-restricted, cryptographically isolated by class, never directly accessible, and only exposed through controlled, augmented overlay contexts to prevent unauthorized extraction, misuse, or cross-domain leakage.

License Notice