Quuipu

SERIE QUUIPU RESEARCH & ARCHITECTURE

WHITEPAPER 06

AI

Il contesto prima dell'intelligenza: integrare gli LLM nell'infrastruttura del Cognitive Web



Whitepaper fondativo · Italiano

Quuipu Edizione 2 · v1

Identità della pubblicazione

Pubblicazione
AI

Serie
Serie Quuipu Research & Architecture

Numero di pubblicazione
Whitepaper 06

Edizione
Quuipu Edizione 2 · v1

Lingua
Italiano

Edizione storica
Cognisphere Edizione 1

Questa pubblicazione succede a Cognisphere Edizione 1. I PDF storici restano immutabili.

Indice

01. Context before intelligence

02. Una configurazione non-LLM può già offrire un'esperienza cognitiva attraverso regole, semantic maps,

03. L'integrazione LLM aggiunge elasticità linguistica, sintesi, trasformazione e capacità di lavorare con richieste

04. Reference architecture

05. AI Gateway

06. Model abstraction

07. Model routing

08. Context packaging

09. RAG as retrieval discipline

10. Memory

11. Prompt minimization

12. Token economics

13. Privacy architecture

14. Local models

15. Mistral as an adapter, not a dependency

16. Other providers

17. Tools and function calling

18. Structured outputs

19. Human control

20. AI transparency

21. Security and prompt injection

22. Evaluation

23. AI failure modes

24. Operating model

25. AI deployment patterns

26. AI invariants

27. The category boundary

28. The final principle

A. AI Gateway request envelope

B. Model Router decision matrix

C. AI risk / control matrix

D. AI-01 · AI is optional

Preamble

Il contesto prima dell'intelligenza: integrare gli LLM nell'infrastruttura del Cognitive Web

> Questa pubblicazione succede a Cognisphere Edizione 1. Quuipu Edizione 2 preserva la tesi storica aggiornando brand e identità di pubblicazione attivi.

Preamble

Context Before Intelligence: Integrating LLMs into Cognitive Web Infrastructure

FOUNDATIONAL THESIS

Prima il contesto, poi l'intelligenza L'LLM estende Quuipu; non lo definisce.

Quuipu does not require an LLM to exist. An LLM can extend Quuipu's capabilities.

Context / Intent / Source / Policy Deterministic Rules, semantic maps, curated representations

Context / Intent / Source / Policy AI Gateway Model Router → Mistral / Other / Local

REFERENCE ARCHITECTURE

Deterministico + AI ↓

Decision: deterministic or AI ↓

AI Gateway ↓

Model Router ↓

Mistral / Other / Local ↓

Validation & Trust ↓

Experience Layer

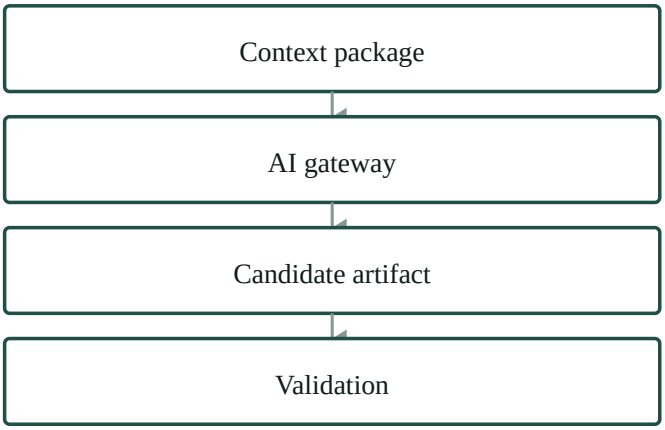


Figure 1 — Flusso di riferimento Quuipu AI.

The model is downstream of context, source authority and policy - and upstream of validation, provenance and human control.

CAPITOLO 01

Context before intelligence

Quuipu non nasce da un modello linguistico. Nasce dall'idea che un'esperienza digitale debba risolvere prima contesto, intento, continuità, significato e fonte, e solo dopo decidere se l'intelligenza generativa aggiunge valore.

La tesi Quuipu does not require an LLM to exist. An LLM can extend Quuipu's capabilities. Questo separa la categoria Cognitive Web dalla categoria chatbot.

Why context first Un modello può produrre linguaggio senza conoscere quale fonte abbia autorità, quale versione sia corretta o quale obiettivo l'utente abbia dichiarato. Quuipu risolve questi vincoli prima dell'inferenza.

AI as optional capability L'AI Gateway è una capability sostituibile e disattivabile. Il sito, la canonical source, la navigation e le trasformazioni deterministiche continuano a esistere senza un provider LLM.

CAPITOLO 02

Una configurazione non-LLM può già offrire un'esperienza cognitiva attraverso regole, semantic maps,

source contracts, context resolution e rappresentazioni curate.

Deterministic runtime Intent mapping, topic routing, source selection, relationship traversal e fallback possono essere deterministici.

Curated transformations Original, Simple o Step-by-step possono utilizzare rappresentazioni editoriali predefinite quando il dominio richiede massimo controllo.

Semantic navigation Explore può seguire un relationship graph canonico senza generazione libera.

Why this matters La capacità di funzionare senza LLM evita lock-in, riduce costi e rende più chiaro che Quuipu è infrastruttura, non una UI conversazionale.

Capability Without LLM With LLM

Understand Curated/simple variants Dynamic simplification within contract

Explore Canonical relationship graph Bounded synthesis over retrieved evidence

Navigation Rules + semantic mapping Natural-language intent interpretation

Fallback Canonical content Same canonical fallback remains

CAPITOLO 03

L'integrazione LLM aggiunge elasticità linguistica, sintesi, trasformazione e capacità di lavorare con richieste

non previste, mantenendo però i confini del runtime.

Extension, not replacement Il modello estende le capability; non sostituisce source authority, context engine, policy, provenance o fallback.

Candidate generation L'output generativo è un candidato da validare contro evidence, claimability e Semantic Integrity.

Bounded tasks Il modello può semplificare, spiegare, sintetizzare, confrontare o strutturare entro un task contract esplicito.

No chatbot requirement Una capability LLM può essere invocata dietro Experience, Understand o Explore senza esporre una chat general-purpose.

CAPITOLO 04

Reference architecture

L'architettura separa il deterministic path dall'AI path. Il Core decide quando una richiesta richiede inferenza generativa e la invia all'AI Gateway.

Core boundary

AI Gateway Media fra runtime e provider: normalizza request, policy, timeout, privacy, logging e response envelope.

Model Router Seleziona un modello sulla base di task, privacy, latency, cost, modality e deployment policy.

Provider adapters Mistral, altri provider o modelli locali sono adapter dietro un contratto comune.

Return path La risposta rientra nel validation layer prima di diventare esperienza.

Layer Owns Must not own

Core Context, state, source, policy Provider-specific semantics

AI Gateway Request policy, privacy, limits Canonical truth

Model Router Model selection Experience authority

Adapter Provider protocol Domain logic

Validation Evidence, schema, integrity Free-form UX rendering

CAPITOLO 05

AI Gateway

L'AI Gateway è il punto di controllo che impedisce al provider di diventare l'architettura.

Normalized contract Il runtime invia un task package indipendente dal provider invece di costruire logica applicativa intorno a una singola API.

Policy enforcement Il gateway applica allowlist di modelli, data class, timeout, max tokens, tool permissions e fallback.

Observability Registra metriche operative e decision trace minimizzato senza richiedere behavioral profiling.

Failure containment Rate limit, provider outage o output invalido degradano verso un altro modello o verso il deterministic runtime.

CAPITOLO 06

Model abstraction

L'astrazione del modello mantiene stabile Quuipu mentre modelli, versioni e provider evolvono.

Capability descriptors Ogni modello può dichiarare modality, context window, structured output, tool support, latency class, privacy class e cost class.

No semantic coupling Il dominio non dovrebbe conoscere nomi di modelli; richiede capability.

Version awareness Model ID e versione fanno parte della provenance quando un output è derivato dall'AI.

Lifecycle Deprecazioni e sostituzioni vengono gestite dal router e dai release gate, non disperse nel codice dell'esperienza.

CAPITOLO 07

Model routing

Routing significa scegliere il modello minimo adeguato al task, non inviare tutto al modello più grande.

Task fit Classificazione, riscrittura breve, sintesi complessa e multimodalità possono richiedere classi diverse.

Latency fit Interazioni inline privilegiano modelli veloci; processi asincroni possono usare modelli più profondi.

Privacy fit Dati sensibili possono richiedere region, provider o local deployment specifici.

Cost fit Budget e token ceiling possono essere parte della policy.

Resilience Router e gateway possono supportare fallback provider o deterministic fallback senza modificare la UI.

Routing dimension Example policy Effect

Task rewrite vs synthesis Choose capability class

Latency inline < asynchronous Prefer faster model

Privacy local-only / EU-region Restrict deployment

Cost token ceiling Small-first / fallback

Resilience provider health Alternate adapter

CAPITOLO 08

Context packaging

Quuipu non consegna al modello tutta la sessione. Costruisce il minimo pacchetto di contesto necessario al task.

Declared intent L'intento esplicito dell'utente è preferibile a inferenze comportamentali.

Relevant state Solo lo state necessario alla decisione corrente entra nel package.

Source excerpts Il retrieval fornisce evidence mirata con identity, locator e version.

Policy envelope Il package include transformation contract, forbidden claims, output schema e disclosure requirements.

Context budget Ogni elemento deve giustificare il proprio costo in token, privacy e rischio.

Declared intent ↓

Relevant state ↓

Authority-filtered retrieval ↓

Evidence excerpts ↓

Transformation contract ↓

Output schema

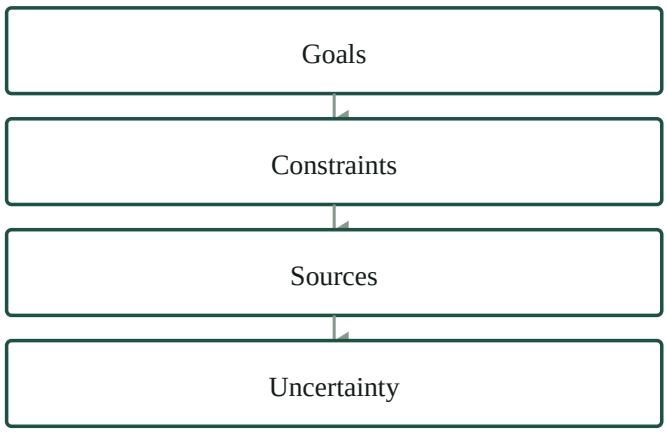


Figure 2 — Context package.

CAPITOLO 09

RAG as retrieval discipline

RAG in Quuipu non è semplicemente 'aggiungere documenti al prompt'. È una pipeline di source resolution, retrieval, evidence selection e provenance binding.

Authority-aware retrieval La ricerca parte da fonti ammesse e authority scope, non dall'intero corpus indiscriminato.

Chunk identity Ogni chunk conserva source identity, version e locator.

Retrieval ranking Semantic relevance è combinata con authority, freshness e scope.

Evidence map I passaggi selezionati vengono associati ai claim che possono supportare.

Post-generation validation La risposta viene verificata contro evidence e policy prima dell'emissione.

Source authority ↓

Retrieval ↓

Chunk identity + version ↓

Evidence ranking ↓

LLM candidate ↓

Claim validation ↓

Provenance binding

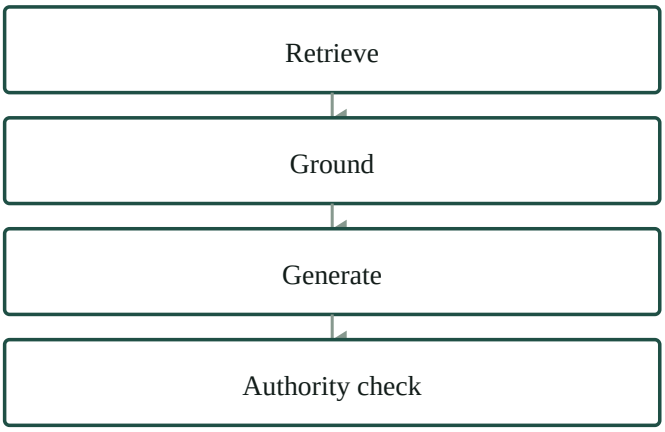


Figure 3 — RAG consapevole dell'autorità.

CAPITOLO 10

Memory

La memoria deve servire continuità senza trasformarsi in profilazione illimitata.

Three memories È utile distinguere session state, explicit continuity memory e domain knowledge memory.

User-declared continuity Preferenze o obiettivi persistenti dovrebbero essere espliciti, revocabili e separati dal comportamento implicito.

Model memory is not canonical memory Una funzione di memoria del provider non sostituisce il memory/state layer governato da Quuipu.

Retrieval over replay Quando possibile, recuperare state strutturato è preferibile a reinviare lunghi transcript.

Expiry TTL, deletion e scope sono parte del contratto di memoria.

Memory class Typical lifetime AI use

Session state Minutes / session Current intent and navigation

Explicit continuity User/org governed Persistent goals or preferences

Domain knowledge Versioned Retrieval source

Provider memory Provider-specific Optional; never canonical

CAPITOLO 11

Prompt minimization

Il prompt più lungo non è necessariamente il prompt migliore. Quuipu mira a inviare soltanto istruzioni e evidence necessarie.

Stable system contract Le regole invarianti sono compatte, versionate e separate dal contenuto.

Structured context JSON o envelope strutturati riducono ambiguità rispetto a transcript concatenati.

No decorative context Informazioni non necessarie vengono escluse anche se potrebbero rendere l'output apparentemente più personalizzato.

Security benefit Meno contesto riduce superficie di leakage e prompt injection.

Economic benefit Meno token riducono costo e latenza.

CAPITOLO 12

Token economics

Il costo AI deve essere una variabile architetturale misurabile, non una sorpresa di produzione.

Cost per cognitive action Misurare token input/output e costo per capability rende confrontabili Experience, Understand e Explore.

Small-first routing Task semplici possono usare modelli piccoli o deterministic logic.

Cache carefully Caching è utile soltanto quando source version, context e policy rendono sicuro il riuso.

Budget ceilings Tenant, sessione o capability possono avere limiti di costo.

Value threshold L'AI viene invocata quando il beneficio atteso supera costo, latenza e rischio.

Cost lever Architecture response Trade-off

Input tokens Prompt minimization + retrieval May require better ranking

Output tokens Structured concise schema Less free prose

Model size Small-first routing Escalation needed for hard tasks

Repeated work Version-safe cache Invalidation complexity

Tenant budget Quota / ceiling Possible deterministic fallback

CAPITOLO 13

Privacy architecture

L'AI path deve rispettare gli stessi confini di privacy del resto di Quuipu.

Data minimization Solo dati necessari al task vengono inviati al provider.

Classification Data classes possono determinare se una richiesta è cloud-eligible, region-bound o local-only.

No raw credentials Token, cookie o segreti host non entrano nel context package.

Provider controls Retention, training use, region e enterprise controls devono essere verificati per provider e piano.

Local option Per alcuni domini, un modello locale può ridurre trasferimenti esterni e aumentare controllo.

CAPITOLO 14

Local models

I modelli locali rendono possibile spostare l'inferenza vicino ai dati o al dispositivo, ma introducono nuovi trade-off.

Benefits Sovranità, latency locale, offline capability e controllo sul lifecycle.

Costs Hardware, aggiornamenti, evaluation, sicurezza e capacity planning diventano responsabilità del deployment.

Hybrid routing Il router può usare local-first e cloud escalation per task più complessi.

Same contracts Local non significa untrusted-by-default né trusted-by-default: provenance, evaluation e policy restano necessarie.

CAPITOLO 15

Mistral as an adapter, not a dependency

Mistral rappresenta un esempio particolarmente rilevante per una strategia europea e ibrida, ma Quuipu non deve dipendere semanticamente da un singolo vendor.

Why relevant Mistral offre API, modelli open-weight e commerciali, strumenti RAG e opzioni di deployment/self-hosting; la documentazione corrente presenta inoltre controlli enterprise e privacy.

Adapter contract L'integrazione dovrebbe vivere dietro un MistralAdapter con mapping di capability, request envelope, errors, usage e provenance.

Routing choice Un modello Mistral può essere scelto quando soddisfa meglio task, cost, privacy o deployment constraints.

No endorsement lock-in Il whitepaper usa Mistral come caso concreto; il Model Router resta aperto a Other e Local.

Mistral integration concern Quuipu abstraction Reason

Model family/version Capability descriptor Avoid domain coupling

API request MistralAdapter Provider isolation

RAG/embeddings Retrieval interface Keep authority-aware retrieval

Privacy/deployment Routing policy Cloud/self-host choices

Usage/cost Gateway telemetry Cross-provider economics

CAPITOLO 16

Other providers

L'apertura a provider multipli riduce dipendenza e permette di ottimizzare capability diverse.

Provider neutrality Il Core non assume API, tokenizer o semantics proprietarie.

Comparative evaluation Modelli vengono valutati su task Quuipu: fidelity, evidence use, structured output, latency, cost e safety.

Policy allowlists Un tenant può autorizzare soltanto determinati provider o region.

Exit strategy Prompt contracts, datasets e evaluation harness devono consentire sostituzione senza riscrivere il prodotto.

CAPITOLO 17

Tools and function calling

Un LLM può orchestrare tool, ma l'accesso alle azioni deve essere più ristretto dell'accesso alla generazione testuale.

Read tools Retrieval e lookup possono essere concessi entro source scope.

Write tools Azioni che modificano dati richiedono autorizzazione, schema, idempotency e spesso conferma umana.

Tool registry Il gateway espone solo tool approvati per la capability corrente.

No ambient agency Il modello non scopre autonomamente nuovi privilegi o tool.

Trace Ogni tool call rilevante entra nel decision trace.

CAPITOLO 18

Structured outputs

Per integrare AI nel runtime, output strutturati sono spesso più utili del testo libero.

Schema-first Il task contract definisce campi, enum, citations/evidence refs e failure state.

Validation Output non conforme viene riparato in modo bounded, ritentato o rifiutato.

Rendering separation Il modello produce semantic payload; l'Experience Layer decide come renderizzarlo.

Safer evolution Schema versioning consente di cambiare modello senza rompere componenti UI.

CAPITOLO 19

Human control

L'AI path resta subordinato a scelte dell'utente e dell'organizzazione.

Visible invocation Quando utile o richiesto, l'utente può sapere che una capability AI è stata invocata.

Mode control L'utente può tornare a Original o a un percorso deterministico.

Operator controls Tenant admin può disabilitare provider, capability, memory o tool.

Escalation Claim ad alta materialità possono richiedere human review.

Override Un override deve produrre effetto reale e tracciabile.

CAPITOLO 20

AI transparency

Trasparenza significa rendere comprensibile il ruolo dell'AI senza attribuirle più autorità di quella che possiede.

AI versus source La UI distingue contenuto canonico da rappresentazione derivata.

Provider versus capability L'utente deve capire la funzione dell'AI; il nome del provider è rilevante quando necessario per policy o disclosure.

Generated content Marking e disclosure possono essere applicati all'emissione senza modificare la canonical source.

EU context Dal 2 agosto 2026 gli obblighi di trasparenza dell'Articolo 50 dell'AI Act si applicano alle categorie previste; le linee guida della Commissione del 20 luglio 2026 chiariscono scope e applicazione.

Legal boundary L'applicabilità concreta dipende da ruolo, sistema e use case e richiede valutazione specifica.

Transparency object User-facing treatment Runtime anchor

AI invoked Disclosure/state where applicable capabilityRef

Derived representation Differentiate from canonical transformationRef

Source support Source/provenance link evidenceRefs

Uncertainty Qualification trustState

Generated content marking Metadata where applicable emissionPolicy

CAPITOLO 21

Security and prompt injection

RAG e tool use ampliano la superficie di attacco. Context-first significa anche trattare fonti e istruzioni come oggetti con livelli di fiducia distinti.

Instruction hierarchy Contenuto recuperato non può ridefinire system policy.

Source sanitization HTML, documenti e metadata possono contenere istruzioni malevole o contenuto non affidabile.

Tool isolation Il modello vede soltanto capability autorizzate.

Output validation URL, action parameters e structured output vengono validati fuori dal modello.

Least privilege Ogni richiesta riceve il minimo accesso necessario.

CAPITOLO 22

Evaluation

La scelta di un modello è una decisione di prodotto verificabile, non una classifica generica di benchmark.

Task suites Dataset rappresentativi misurano Simple, Step-by-step, synthesis, extraction, classification e conflict handling.

Semantic fidelity WP04 Semantic Integrity fornisce invarianti per verificare che la trasformazione non alteri il significato.

Trust tests WP05 aggiunge provenance, uncertainty, policy, disclosure e fallback.

Cost and latency Quality viene valutata insieme a token, time-to-first-token e total latency.

Regression gates Nuove versioni di modello entrano in produzione solo dopo comparative evaluation.

Evaluation axis Question Metric/example

Fidelity Did meaning change? Invariant pass rate

Grounding Are claims supported? Claim support ratio

Latency Is interaction responsive? TTFT / total latency

Cost Is routing efficient? Cost/action

Trust Are limits visible? Fallback/disclosure correctness

CAPITOLO 23

AI failure modes

L'architettura deve assumere che il modello possa sbagliare anche quando il provider è disponibile.

Hallucination Claim non supportati vengono intercettati da evidence/claimability controls.

Instruction drift Output fuori task contract viene rifiutato o ritentato.

Provider outage Router prova un adapter alternativo o deterministic fallback.

Context overflow Packaging riduce evidence o cambia strategia invece di troncare silenziosamente elementi materiali.

Model deprecation Lifecycle registry e evaluation consentono migrazione controllata.

CAPITOLO 24

Operating model

Portare LLM in Quuipu richiede responsabilità operative chiare.

AI capability owner Definisce task, schema, evaluation e SLO.

Model governance Approva provider, versioni, data classes e routing policy.

Domain owner Definisce source authority e materiality.

Engineering Gestisce gateway, adapters, security, observability e cost controls.

Trust/compliance Definisce disclosure, logging, retention e review requirements.

CAPITOLO 25

AI deployment patterns

No-AI Solo deterministic runtime e contenuti curati.

Cloud AI Gateway verso provider gestiti.

EU-region cloud Routing vincolato a region e provider policy.

Self-hosted Inference in infrastruttura controllata dall'organizzazione.

Hybrid Local/small-first con escalation verso cloud o modelli più capaci.

Pattern Where inference runs Best fit

No-AI None Maximum determinism

Managed cloud Provider API Fast adoption

Region-bound Selected cloud region Residency policy

Self-hosted Org infrastructure Sovereignty/control

Hybrid Local + cloud Balanced cost/privacy/capability

CAPITOLO 26

AI invariants

Gli invarianti impediscono che l'aggiunta di un LLM trasformi Quuipu in un chatbot opaco o in un sistema model-centric.

AI is optional Il Core rimane valido senza LLM.

Context precedes inference Intent, source, scope e policy vengono risolti prima della generazione.

Model is replaceable Provider e versioni restano dietro adapter.

Source outranks model L'LLM non diventa fonte di verità.

Minimum necessary context Il package contiene solo dati necessari.

Fallback survives AI Il deterministic path resta disponibile.

CAPITOLO 27

The category boundary

La differenza strategica non è 'Quuipu usa AI?'. È dove l'AI viene collocata nell'architettura.

Chatbot pattern Il modello è il punto di ingresso e la conversazione è il prodotto.

L'esperienza cognitiva è il prodotto; il modello è una capability invocata quando utile.

Web-native L'utente resta dentro il sito, le sue fonti e i suoi percorsi.

Governed intelligence Context, Semantic Integrity e Trust Architecture delimitano ciò che il modello può fare.

Market message Quuipu is not another chatbot. It is Cognitive Web infrastructure that can use AI without becoming dependent on it.

Chatbot-centric Quuipu-centric

Conversation is the primary interface Website experience remains primary

Model holds conversational context Runtime packages bounded context

Model often selects knowledge Authority-aware retrieval selects sources

Provider behavior shapes product Provider sits behind adapter

Failure often becomes bad answer Failure can return to canonical web

CAPITOLO 28

The final principle

Quuipu AI definisce un modo di integrare modelli generativi senza consegnare loro l'identità del prodotto, la fonte di verità o il controllo dell'esperienza.

Before intelligence Risolvere contesto, intento, fonte, scope e policy.

Use intelligence Invocare il modello minimo adeguato quando aggiunge valore.

After intelligence Validare evidence, semantic fidelity, trust state e provenance.

Keep control Consentire fallback, override e sostituzione del modello.

The thesis Quuipu does not require an LLM to exist. An LLM can extend Quuipu's capabilities.

CAPITOLO A

AI Gateway request envelope

Field Example Purpose

taskType simplify / synthesize / classify Capability routing

context declared intent + bounded state Task relevance

evidence SourceRef + excerpts Grounding

policy allowed claims/tools/providers Governance

privacyClass cloud / region / local Data routing

budget max input/output/cost Economics

outputSchema versioned schema Validation

trace request + policy IDs Auditability

CAPITOLO B

Model Router decision matrix

Condition Preferred route Fallback

Deterministic rule sufficient No model Canonical/curated

Low-complexity text transform Small/fast model Deterministic representation

Complex synthesis Capable model Reduced-scope answer

Sensitive local-only data Local model No-AI/refusal

Provider unavailable Alternate provider Deterministic fallback

Evidence insufficient No generation Original / clarify

CAPITOLO C

AI risk / control matrix

Risk Control Safe outcome

Hallucination Evidence + claimability gate Block unsupported claim

Prompt injection Instruction hierarchy + sanitization Ignore hostile instructions

Context leakage Minimization + data class Exclude unnecessary data

Provider lock-in Adapters + eval harness Replace model/provider

Cost runaway Budget + routing Small-first/fallback

Model drift Versioned evaluation Release gate

Hidden AI role Disclosure policy Visible invocation where needed

Tool overreach Allowlisted tools + auth Bounded action

CAPITOLO D

AI-01 · AI is optional

AI-02 · Context before inference Intent, scope, source and policy precede generation.

AI-03 · Source before model The model never becomes the canonical source.

AI-04 · Provider abstraction Provider protocol remains outside domain logic.

AI-05 · Minimum context Only task-necessary context is packaged.

AI-06 · Authority-aware RAG Retrieval respects source authority and version.

AI-07 · Candidate output Generative output is validated before experience emission.

AI-08 · Model-aware provenance Model and version are traceable for derived output.

AI-09 · Cost is bounded Token and model economics are policy inputs.

AI-10 · Privacy routes inference Data class can constrain provider, region or local execution.

AI-11 · Human control survives Users/operators can override, disable or fall back.

AI-12 · Deterministic fallback survives AI failure does not remove access to the canonical experience.

APPENDIX E

Relationship to the Quuipu series Whitepaper Contribution to Quuipu AI

01 · The Cognitive Web Defines context-aware experience as the category.

02 · Architecture Provides Core, Host, Experience and state boundaries.

03 · Quuipu Web Shows where AI may enhance website experience without replacing it.

04 · Semantic Integrity Constrains semantic transformation and evidence-backed claims.

05 · Trust Architecture Adds provenance, explainability, bounded intelligence and human control.

06 · Quuipu AI Places LLMs behind context, routing, validation and fallback.

REFERENCES

Selected external references European Commission. Guidelines on transparency obligations for providers and deployers of AI systems, 20 July 2026. The Commission states that Article 50 transparency obligations apply from 2 August 2026. Mistral AI documentation. Current documentation describes API access, RAG and embeddings, model families, workspaces and admini-

stration, privacy controls, and deployment options. These references are used only to ground the vendor example; Quuipu's architecture remains provider-neutral. Architecture note. Product capabilities, model names, licensing, pricing and provider controls evolve. Production routing policy should therefore rely on current provider documentation and versioned evaluation rather than static whitepaper assumptions.

Context Before Intelligence: Integrating LLMs into Cognitive Web Infrastructure

CONTEXT BEFORE INTELLIGENCE. AI EXTENDS THE SYSTEM; IT DOES NOT DEFINE IT.