

HCL DOMINO 2026 (14.5.1 FP1)

DominoIQ

Training Course Support

Overview • Prerequisites • System Constraints
Integration • Configuration • Application Use Cases

Audience: Domino Administrators & Application Developers

Version: HCL Domino 14.5.1 / Fix Pack 1 (released July 2026)

Table of Contents

Table of Contents	2
1. Presentation of DominoIQ	3
1.1 Key idea	3
1.2 Two operating modes	3
1.3 What's new in 14.5.1 FP1	3
2. Prerequisites	3
2.1 Domain / software prerequisites	3
2.2 Skills and roles	4
3. System Constraints	4
3.1 Supported platforms	4
3.2 Other constraints to plan for	5
4. Integration into an Existing Domino Domain	5
4.1 Integration principles	6
4.2 Typical rollout path	6
5. Configuration	6
5.1 Step 1 - Directory Profile	7
5.2 Step 2 - dominoiq.nsf database setup	7
5.3 Local vs. Remote configuration document	7
5.4 Enabling RAG (14.5.1 FP1)	8
6. Application Use Cases	8
6.1 Messaging (Notes / Verse) - quick overview	8
6.2 Custom application use case - Knowledge Base Assistant	8
6.3 Custom application use case - Intake Triage	9
7. Summary	9
Key takeaways	9
8. Sample Codes	9
Resume An Email Thread	9
Check A Command Document Is Available	11
Basic Text Generation	11
List All Available Command Documents	12
Email Summary Agent	13

1. Presentation of DominoIQ

DominoIQ is the generative-AI capability introduced with HCL Domino 14.5 and extended in 14.5.1 / Fix Pack 1 (FP1). It embeds a large language model (LLM) inference engine directly inside the Domino server process, exposing AI features to Notes, Verse and iNotes users, and to Domino applications, without sending company data to an external cloud AI service.

1.1 Key idea

On a server configured for DominoIQ, the Domino server (Dbserver process) starts an AI inference engine listening on a configured local port alongside the other Domino server tasks. Two LotusScript classes - NotesLLMRequest and NotesLLMResponse - (and equivalent Java classes, respectively LLMReq and LLMRes) let developers send a prompt to the engine and read back a completion, synchronously or as a stream, from inside any NSF application.

1.2 Two operating modes

- **Local mode:** the DominoIQ server hosts the inference engine itself (llama.cpp-based), running a GGUF model file on local NVIDIA GPU hardware.
- **Remote mode:** the Domino server has no local inference engine; it forwards LLM requests to an external endpoint (another DominoIQ server, or a third-party OpenAI-compatible / Ollama endpoint). GPU-less servers can still offer DominoIQ to their users this way.

1.3 What's new in 14.5.1 FP1

- **Retrieval-Augmented Generation (RAG):** Domino documents are vectorized (via updall -w) into a local vector database and used as grounding context for answers - with normal Domino ACLs enforced on retrieval.
- **RAG in Remote mode (new in FP1):** the LLM and embedding models can now be hosted remotely, while the vector database itself always stays local to the DominoIQ server.
- **Optional Guard Model:** a secondary, smaller model that screens prompts and/or responses for unsafe or out-of-policy content.

Note: DominoIQ is a feature of the core Domino server platform, distinct from HCL Domino Volt/Leap or the Domino AppDev Pack. It is configured through the Domino Directory and a dedicated database, dominoiq.nsf.

2. Prerequisites

Confirm the following organisational and technical prerequisites before enabling DominoIQ in a domain.

2.1 Domain / software prerequisites

- HCL Domino server at 14.5.0 or later (this course targets 14.5.1 FP1) on at least one server that will act as the DominoIQ server.
- Domino Directory design upgraded to the 14.5+ template (pubnames.ntf), which exposes the "Domino IQ" tab on the Directory Profile.

- An Administration server designated for DominoIQ - it holds the primary replica of dominoiq.nsf that all other DominoIQ servers replicate from.
- For mail/client features: mail files upgraded to the Mail 14.5 template (mail145.ntf) to expose Summarize / Reply-with-AI actions in Notes; HCL Verse on-premises 3.2+ for the browser client, enabled with the VOP_GK_FEATURE_301 notes.ini setting.
- A downloaded LLM in GGUF format (from the My HCLSoftware Portal or another trusted source) for Local mode; or the URL/credentials of a reachable external inference endpoint for Remote mode.

2.2 Skills and roles

- Domino administrator rights on the Directory and on the target servers, to edit the Directory Profile and server documents.
- Designer/developer skills in LotusScript or Java for teams building custom calls to DominoIQ (NotesLLMRequest / NotesLLMResponse).
- Basic familiarity with LLM concepts (prompt, system prompt, context window, embeddings) for whoever authors System Prompt and Command documents.

3. System Constraints

DominoIQ Local mode has real hardware and OS requirements. Remote mode relaxes the hardware requirement on the Domino server but shifts data-flow and network considerations to the remote endpoint.

3.1 Supported platforms

Constraint: Local mode is only supported on 64-bit Windows or Linux Domino servers with a physical NVIDIA GPU. There is no macOS support and no pure-CPU fallback for Local mode.

Requirement	Local mode	Remote mode
Operating system	Windows Server or Linux, 64-bit only	Any OS supported by Domino
CPU / RAM (minimum)	4 CPUs, 8 GB RAM	Standard Domino sizing
GPU	NVIDIA GPU, ≥ 8 GB VRAM, compute capability ≥ 5.2 (≥ 8.0 recommended for production)	Not required on the Domino server itself
GPU driver / toolkit	NVIDIA driver compatible with the CUDA 12.x Toolkit	N/A
Model format	GGUF, stored under data/llm_models	Managed by the remote endpoint

Requirement	Local mode	Remote mode
Models per server	One active model per DominoIQ server instance	Depends on the remote endpoint
Network	Local loopback to inference port (default 8080, or 8443 for TLS)	Outbound HTTPS to the remote endpoint required

3.2 Other constraints to plan for

- Windows servers can use both GPU memory and up to half of system RAM as “Shared RAM”, which eases sizing for small pilots but still requires a GPU.
- The RAG vector database is always stored locally on the DominoIQ server, even in Remote mode - plan local disk space for vectorized NSF content.
- The Notes client caches the list of DominoIQ servers and the available commands; changes to the Directory Profile or to dominoiq.nsf require a client restart to take effect.
- A DominoIQ server runs a single model at a time; supporting several use cases with different models means dedicating separate DominoIQ servers, or using Remote mode against a multi-model endpoint.
- An optional Guard Model runs as an additional inference process on its own port (default 8444) and consumes additional GPU/CPU resources.

4. Integration into an Existing Domino Domain

DominoIQ is designed to be added incrementally to a running domain: it introduces one new database (dominoiq.nsf), one new Directory Profile tab, and one or more dedicated servers - without requiring changes to existing applications unless they choose to call the new APIs.

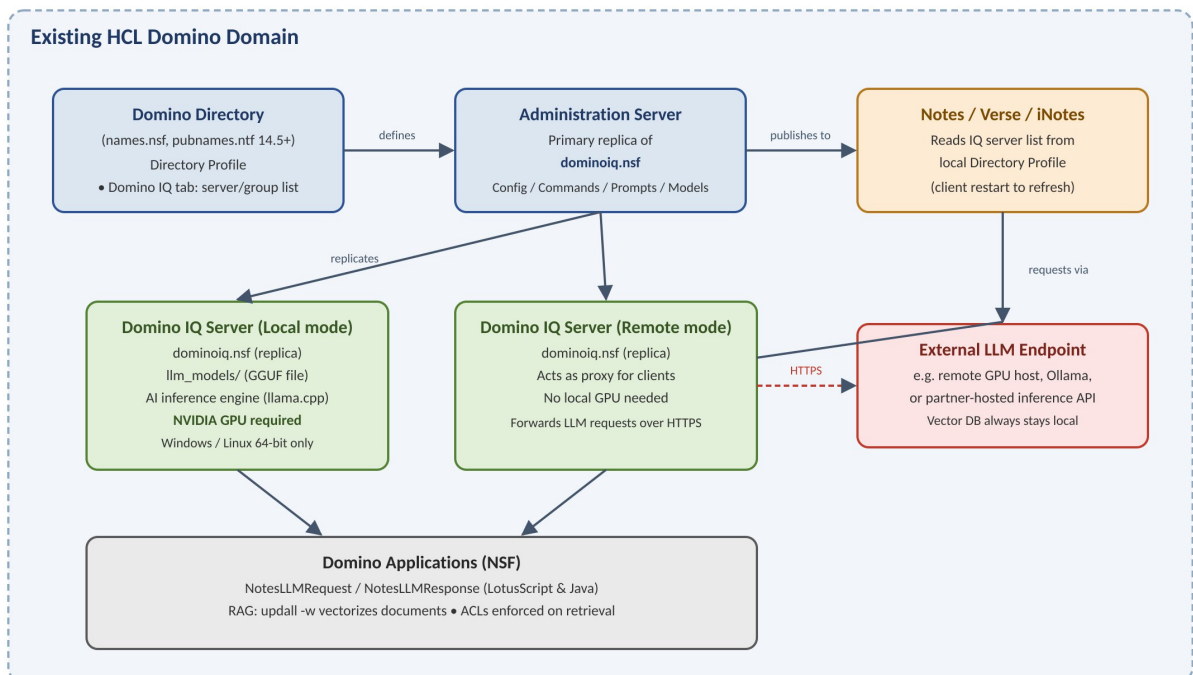


Figure 1 – DominoIQ integration into an existing Domino domain (Local and Remote modes)

4.1 Integration principles

- No impact on servers that are not designated as DominoIQ servers - they continue to operate unchanged.
- dominoiq.nsf is created automatically the first time a server starts with DominoIQ enabled; it replicates like any other Domino database using existing replication schedules and connection documents.
- One or more DominoIQ servers can be declared per domain; a single Administration server centralises configuration to avoid replication conflicts.
- Existing mail routing, directory, and security infrastructure (OIDC, SAML, ID vault, TLS) is reused as-is - DominoIQ does not require a parallel identity or network stack.
- Client-side, Notes/Verse/iNotes (! iNotes is removed in 14.5.x) pick up DominoIQ availability from the user's Directory Profile the next time they connect to their home mail server; a client restart may be required to see new servers or commands.

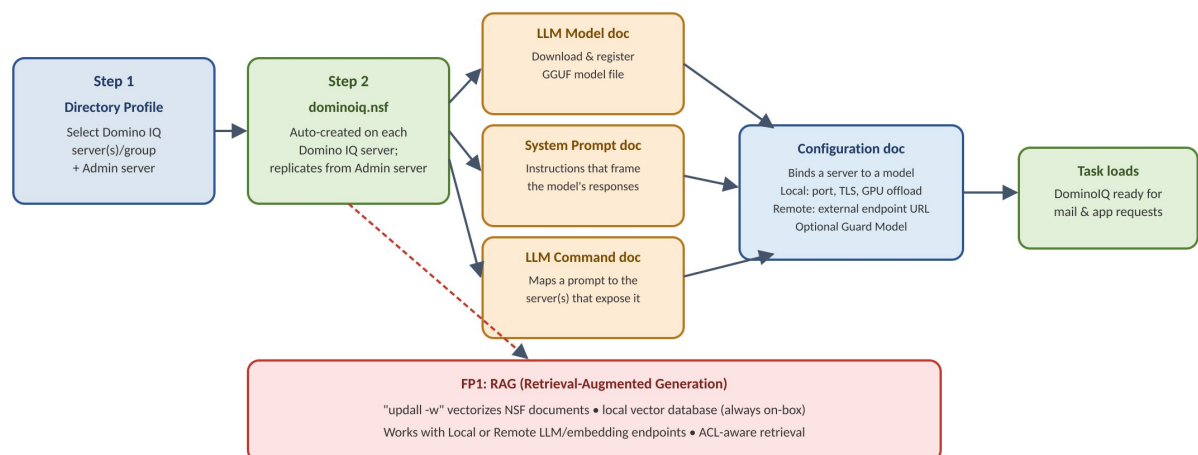
4.2 Typical rollout path

1. Pilot: enable DominoIQ Local mode on one non-production server with a small GGUF model to validate GPU sizing and user acceptance.
2. Directory rollout: upgrade the Domino Directory design and publish the Directory Profile so pilot users see the feature.
3. Scale out: add further DominoIQ servers (Local for GPU-equipped hubs, Remote for branch servers without a GPU) pointing at a shared or per-site model strategy.
4. Application enablement: extend selected NSF applications with NotesLLMRequest/NotesLLMResponse calls, and enable RAG on knowledge-base databases once the pilot is validated.

Note: Because dominoiq.nsf and the Directory Profile changes are additive, DominoIQ can be piloted on a subset of servers/users without impacting the rest of an existing, live Domino domain.

5. Configuration

Configuration happens in two places: the Domino Directory (domain-wide declaration of DominoIQ servers) and the dominoiq.nsf database (per-server and per-feature settings).



5.1 Step 1 - Directory Profile

- Upgrade the Domino Directory design to the current template, then run Actions > Edit Directory Profile.
- On the Domino IQ tab, select the server(s) or group running DominoIQ, and choose the Administration server for DominoIQ.

5.2 Step 2 - dominoiq.nsf database setup

- **LLM Model document:** registers a GGUF model; DominoIQ can auto-download and manage the model file under data/llm_models. (for instance hi4 and Llama 3.x that support paraphrasing : https://help.hcl-software.com/domino/14.5.0/admin/conf_security_considerations_for_iq.html, paraphrasing option is default and Domino domain wide and is available with Domino 14.5.1).
- **System Prompt document:** defines the instructions given to the model to frame how it should answer (tone, scope, format).
- **LLM Command document:** maps a System Prompt to the server(s) that will expose it, optionally overriding model parameters.
- **Configuration document:** binds a Domino server to a model and an inference engine - Local (port, TLS via Certstore, CPU/GPU offload tuning) or Remote (external endpoint URL and credentials) - with an optional Guard Model tab.

5.3 Local vs. Remote configuration document

Setting	Local mode	Remote mode
Inference target	localhost, default port 8080 (8443 with TLS)	External endpoint URL
TLS	Optional, via Domino Certstore	Depends on the remote endpoint
Performance tuning	Concurrent CPUs, GPU offload, layers offloaded	Not applicable on this server
Guard Model	Optional, own port (default 8444)	Optional, if the endpoint supports it

5.4 Enabling RAG (14.5.1 FP1)

- Identify the NSF databases to expose as a knowledge base and run updall -w to vectorize their documents into the local vector database.
- Configure the embedding model and, since FP1, optionally point it and the LLM at a remote endpoint while the vector store itself remains local.
- Retrieval automatically respects each document's existing Domino ACLs, so a user only ever gets answers grounded in content they are authorized to read.

Note: Always make configuration changes on the designated Administration server for DominoIQ, then let normal replication push them to the other DominoIQ servers, to avoid replication conflicts in dominoiq.nsf.

6. Application Use Cases

DominolQ can be consumed in two complementary ways: through ready-made mail client features, and through custom calls from any Domino application.

6.1 Messaging (Notes / Verse) - quick overview

- **Summarize an email thread:** one click condenses a long thread into key points and action items.
- **Reply with AI:** drafts a reply based on the message content and a chosen tone/instruction.
- **Requires:** mail145.ntf template, a DominoIQ server declared in the Directory Profile, and (for Verse) the VOP_GK_FEATURE_301 setting.

6.2 Custom application use case - Knowledge Base Assistant

Scenario: an HR or IT knowledge base (an existing NSF of policies and procedures) needs a natural-language “ask a question” assistant, without moving data to the cloud.

1. Enable RAG on the knowledge-base NSF and run upcall -w to vectorize its documents.
2. Create a System Prompt instructing the model to answer strictly from retrieved context and to say when it does not know.
3. Add an LLM Command exposing this prompt on the relevant DominoIQ server(s).
4. In a simple XPages or Notes Client action, call session.CreateLlmRequest(), pass the user's question, and display the NotesLLMResponse content together with links to the source documents.
5. Because retrieval respects ACLs, two users with different access rights automatically get answers grounded only in the documents each is allowed to see.

6.3 Custom application use case - Intake Triage

Scenario: a helpdesk or requests database receives free-text submissions that must be classified and routed.

1. Define a Command whose System Prompt asks the model to classify a submission into one of a fixed set of categories and extract a priority.
2. On document save, a LotusScript agent calls NotesLLMRequest with the submission text and the Command name.
3. The agent parses the NotesLLMResponse content and sets category/priority fields, then routes the document with standard Domino workflow (mail, views, @formulas).

Note: In both application scenarios, DominoIQ only adds an inference call around existing Domino data and workflow - the surrounding application (forms, views, ACLs, workflow) stays entirely standard Domino design.

7. Summary

DominolQ brings on-premises generative AI - including RAG over existing NSF content - into the Domino server itself, in Local or Remote mode, configured through the Directory Profile and dominoiq.nsf. It integrates without disruption into an existing domain, and can be consumed immediately through mail client features or through simple LotusScript/Java calls in custom applications.

Key takeaways

- Local mode needs a 64-bit Windows/Linux server with an NVIDIA GPU; Remote mode removes that constraint from the Domino server but requires a reachable external endpoint.
- Configuration is domain-wide (Directory Profile) plus per-server (dominoiq.nsf: Model, System Prompt, Command, Configuration documents).
- RAG (14.5.1 FP1) keeps the vector database local and ACL-aware, even when the LLM itself runs remotely.
- Rollout can start small (single pilot server, single use case) and expand server by server, use case by use case.

8. Sample Codes

Resume An Email Thread

Sub Initialize

On Error Goto processError

Dim session As New NotesSession

Dim workspace As New NotesUIWorkspace

Dim mailDoc As NotesUIDocument

Set mailDoc = workspace.CurrentDocument

Dim sMailThread As String

mailDoc.Editmode = True

sMailThread = mailDoc.Fieldgettext("Body")

Dim llmreq As NotesLLMRequest

```

Dim llmres As NotesLLMResponse

Set llmreq = session.CreateLLMRequest()

' 1er paramètre : nom du serveur DominoIQ ("" = routage auto)
' 2e paramètre : nom du document "Command" défini dans dominoiq.nsf
' 3e paramètre : le prompt utilisateur

Set llmres = llmreq.Completion("DominoIQ server/Org", "StdSummarizeEmailThread",
sMailThread)

If (llmres.FinishReason = LLM_FINISH_REASON_STOP) Then

    Call mailDoc.FieldSetText("Summary", llmres.Content)

    Call mailDoc.save()

End If

Exit Sub

processError:

    MessageBox(Error$)

    Exit Sub

End Sub

```

Check A Command Document Is Available

```

Dim llmreq As NotesLLMRequest

Set llmreq = session.CreateLLMRequest()

If llmreq.IsCommandAvailable("DominoIQ server/Org", "StdSummarizeEmailThread", True)
Then

    Set llmres = llmreq.Completion("DominoIQ server/Org", "StdSummarizeEmailThread",
userText)

Else

    MessageBox "Commande non configurée sur ce serveur DominoIQ."

```

End If

Basic Text Generation

Sub Initialize

Dim session As New NotesSession

Dim llmReq As NotesLLMRequest

Dim llmRes As NotesLLMResponse

Dim prompt As String

Dim replyText As String

' Initialize the Domino IQ LLM Request object

Set llmReq = session.CreateLLMRequest()

' Define your prompt

prompt = "Summarize the key benefits of upgrading to HCL Domino 14.5 in three bullet points."

' Send the request synchronously to the Domino IQ server

Set llmRes = llmReq.Completion(prompt)

' Check if the response was successful

If Not (llmRes Is Nothing) Then

replyText = llmRes.GetText()

MsgBox "Domino IQ Response: " & replyText, 64, "Success"

Else

MsgBox "Failed to get a response from Domino IQ.", 16, "Error"

End If

End Sub

List All Available Command Documents

Sub Initialize

On Error Resume Next

Dim session As New NotesSession

Dim llmreq As NotesLLMRequest

Dim availCmds As Variant

Dim displayMsg As String

Set llmreq = session.CreateLLMRequest()

availCmds = llmreq.GetAvailableCommands("DominolQ server/Org")

If (Not Isempty(availCmds) And Len(availCmds(0)) > 0) Then

Forall cmd In availCmds

displayMsg = displayMsg & cmd & Chr(10)

End Forall

MessageBox displayMsg, MB_OK, "Commandes disponibles"

Else

MessageBox "Aucune commande disponible."

End If

End Sub

Email Summary Agent

Sub Initialize

Dim session As New NotesSession

Dim db As NotesDatabase

Dim dc As NotesDocumentCollection

Dim doc As NotesDocument

Dim llmReq As NotesLLMRequest

```

Dim llmRes As NotesLLMResponse
Dim prompt As String

Set db = session.CurrentDatabase
Set dc = db.UnprocessedDocuments
Set doc = dc.GetFirstDocument()

If doc Is Nothing Then Exit Sub

' Initialize Domino IQ
Set llmReq = session.CreateLLMRequest()

' Construct a context-driven prompt using document items
prompt = "Analyze the tone of the following text and write a brief summary: " &
doc.GetItemValue("Body")(0)

' Execute the query
Set llmRes = llmReq.Completion(prompt)

If Not (llmRes Is Nothing) Then
    ' Save the result back into a rich text or text field in the document
    Call doc.ReplaceItemValue("AI_Summary", llmRes.GetText())
    Call doc.Save(True, False)
    MsgBox "Document successfully updated with Domino IQ summary!", 64, "Done"
End If
End Sub

```