IKE Coordinate System - Knowledge Engineer's Guide

Table of Contents

*A **practical guide for knowledge engineers and business analysts to** understand the IKE Coordinate System (Integrated Knowledge Ecosystem). Learn how coordinates control what you see when browsing, searching, and editing knowledge.*

Note: This coordinate system is based on the HL7 Terminology Knowledge Architecture (TINKAR) specification. Technical references use "Tinkar" in code for backward compatibility.

1. About This Guide

For: Knowledge Engineers, Business Analysts, Content Managers, Clinical Informaticists

You'll Learn:

- What coordinates are and why they matter
- The five coordinate types and when to use each
- How to configure coordinates for common tasks
- Best practices for terminology work

You Will NOT Find: Code examples, programming syntax, or API documentation (see the Developer's Guide for those)

2. The Five Coordinate Types

Type	Controls	Your Question
STAMP	Time, status, modules, paths	"When and where am I looking?"
Language	Descriptions, translations, dialects	"What language do I want?"
Logic	Classification, reasoning	"Stated or inferred relationships?"
Navigation	Hierarchies, sorting	"How do I browse?"
Edit	Authoring, attribution	"Who am I and where does my work go?"

Think of coordinates as **viewing filters** that determine what content you see and how it's displayed.

3. Document Sections

4. Understanding Coordinates

4.1. What Are Coordinates?

A **coordinate** is a set of specifications that tells the knowledge system what content to show you and how to display it. Think of coordinates like viewing filters on a website:

- Change language: English to Spanish
- Filter by date: Show content from last month
- Filter by status: Show active items only
- Sort results: Alphabetical or by relevance

IKE coordinates work the same way—they control your view of terminology content.

4.2. Why Coordinates Matter

Knowledge engineers face these challenges daily:

Time Problem: Which version of "Pneumonia" should you see—today's, last month's, or from last year?

Language Problem: Should "Diabetes mellitus" display in English or Spanish? Preferred term or fully qualified name?

Hierarchy Problem: Should you see the hierarchy as authors wrote it (stated) or as the classifier

computed it (inferred)?

Authoring Problem: When you create new concepts, who is recorded as the author, and which module do they go into?

Coordinates solve all these problems by explicitly specifying your viewing context.

4.3. About IKE and TINKAR

The **IKE Coordinate System** is the open-source framework for managing complex knowledge at scale, based on the HL7 Terminology Knowledge Architecture (TINKAR) specification.

- **TINKAR** = Formal HL7 specification and technical foundation
- **IKE** = Open-source ecosystem and community implementation

Throughout this guide, we refer to the **IKE Coordinate System** conceptually, while technical artifacts may reference "TINKAR" for backward compatibility.

4.4. The Five Coordinate Types

Coordinate	What It Controls
STAMP	**When** you're looking (time, status, modules, development paths)
Language	**What language** and description types to display
Logic	**How** concepts are classified (stated vs. inferred relationships)
Navigation	**Which hierarchies** to browse and how to sort them
Edit	**Who you are** and where your authoring work goes

4.5. How Coordinates Work Together

All five coordinate types combine to create a complete "view" of your knowledge base.

Example Clinical Browse Configuration:

- STAMP: Active only, Latest, Production path
- Language: US English preferred terms
- Logic: Inferred relationships
- Navigation: Inferred hierarchy, alphabetical sorting
- Edit: Not applicable (read-only)

Result: Clinicians see complete, current, active concepts with readable names in a logical hierarchy.

Example Content Authoring Configuration:

- STAMP: Both active/inactive, Latest, Development path

- Language: US English, fully qualified names
- Logic: Stated relationships (while authoring)
- Navigation: Stated hierarchy
- Edit: Your name, Local Extension module, Development path

Result: You see everything needed for authoring, with clear attribution and proper content organization.

4.6. Key Benefits

Consistency: Every operation uses the same coordinate—no mixed views.

Reproducibility: Same coordinates = same results. Critical for collaboration.

Transparency: Coordinates make assumptions explicit—you know exactly what you're viewing.

Flexibility: Change perspective instantly by changing coordinates.

4.7. When You Interact with Coordinates

You interact with coordinates during:

- **Browsing/Searching** - Coordinates determine what you can see
- **Quality Assurance** - Compare versions, check translations, validate hierarchies
- **Content Authoring** - Coordinates track you as author and organize your work
- **Release Management** - Verify content before promoting to production

4.8. What's Next

The following sections explain each coordinate type:

- STAMP - Time, status, and versioning
- Language - Multilingual descriptions
- Logic - Classification and reasoning
- Navigation - Hierarchies and browsing
- Edit - Authoring and workflows

Then we'll cover practical scenarios and best practices.

5. STAMP Coordinates

5.1. Overview

STAMP coordinates answer: **"When am I looking, and from which branch?"**

STAMP = *S*tatus + *T*ime + *A*uthor + *M*odule + *P*ath

Every piece of content carries STAMP metadata. STAMP coordinates tell the system which versions to show you.

5.2. The Five Components

5.2.1. Status: Active vs. Inactive

- **Active** - Current, should be displayed
- **Inactive** - Retired, deprecated, deleted

When to use:

- Active only → End users, clinical systems (most common)
- Both → Authoring, QA, historical analysis
- Inactive only → Finding retired concepts (rare)

5.2.2. Time: When Are You Looking?

- **Latest** - Current state (95% of use cases)
- **Specific Date** - Historical view for analysis or troubleshooting

Example: Set time to "2 weeks ago" to see what content looked like then, compare with today to find changes.

5.2.3. Author: Who Made Changes?

Tracks who created or modified content (tracked but not filtered):

- For audit trails and accountability
- For quality reviews
- For change attribution

Note: Author is informational—you typically don't filter by it.

5.2.4. Module: Organizational Grouping

Modules group content by ownership:

- SNOMED CT Core
- US Extension

- Local Hospital Extension

Options:

- Include specific modules only
- Exclude specific modules
- Prioritize modules (when conflicts exist)

5.2.5. Path: Development Branches

Paths represent parallel development lines:

- **Production** - Released content
- **Development** - Work in progress
- **Staging** - Testing before release
- **Sandbox** - Experiments

Key Point: Content is visible on a path if created there OR inherited from parent path before branching.

5.3. Common Configurations

Use Case	STAMP Settings
Clinical users	Active only, Latest, Production
Content authors	Both active/inactive, Latest, Development
QA reviewers	Both active/inactive, Latest, Staging
Historical analysis	Both, Specific date, Production

5.4. Decision Guide

Status:

- Active only → End users
- Both → Authoring, QA, analysis

Time:

- Latest → 95% of cases
- Historical → Analysis, troubleshooting

Module:

- All modules → General work
- Specific modules → Focused reviews

Path:

- Production → End users
- Development → Authoring
- Staging → QA/testing

6. Language Coordinates

6.1. Overview

Language coordinates answer: **"What language and description type do I want?"**

A single concept can have many descriptions (multiple languages, types, dialects). Language coordinates specify your preferences.

6.2. The Four Components

6.2.1. 1. Language

Natural language: English, Spanish, French, etc.

6.2.2. 2. Description Types

- **Preferred Term** - Main name ("Pneumonia")
- **Fully Qualified Name** - With disambiguation ("Pneumonia (disorder)")
- **Synonym** - Alternative name ("Lung infection")
- **Definition** - Full explanation

6.2.3. 3. Dialect Preferences

Regional variants:

- US English vs. GB English ("Color" vs. "Colour")
- Mexican Spanish vs. Spain Spanish

6.2.4. 4. Module Preferences

When multiple descriptions exist, which source wins (Local > National > Core).

6.3. Cascading Languages

Specify fallback order:

1. Spanish (primary)

2. English (fallback)
3. Any language (last resort)

Why: Translation gaps are common. Fallback ensures users always see something.

6.4. Common Configurations

Audience	Language Settings
US Clinicians	English, Preferred Terms, US dialect
Spanish Patients	Spanish → English fallback, Preferred Terms
Technical Staff	English, Fully Qualified Names (disambiguation)
Help Systems	English, Definitions

6.5. Decision Guide

Description Type:

- Preferred Terms → End users, general browsing
- Fully Qualified Names → Technical staff, disambiguation needed
- Synonyms → Patient-friendly interfaces
- Definitions → Help systems, training

Dialect:

- Set based on user location
- Always provide fallback

Cascading:

- Always use fallback languages for incomplete translations

7. Logic Coordinates

7.1. Overview

Logic coordinates answer: **"Should I see stated or inferred relationships?"**

They control whether you see relationships as authored (stated) or as computed by the classifier (inferred).

7.2. Stated vs. Inferred

7.2.1. Stated Relationships

What: Relationships you explicitly authored

Characteristics:

- Direct author control
- Sparse (only necessary relationships)
- What you see in editing tools

Example: "Bacterial pneumonia IS-A Pneumonia" (what you wrote)

7.2.2. Inferred Relationships

What: Relationships computed by the classifier

Characteristics:

- Automatically generated
- Complete (all logical implications)
- What you want for browsing

Example: Classifier adds: "Bacterial pneumonia IS-A Lung disease" and "Bacterial pneumonia IS-A Infectious disease" (computed from logic)

7.3. When to Use Each

Activity	Stated	Inferred
Authoring	✓	
Browsing		✓
Searching		✓
End Users		✓
QA Review	✓	✓ (both)
Debugging	✓	✓ (both)

Key Rule: Author with stated, browse with inferred.

7.4. Classification

Classification is the process where a reasoning engine:

1. Reads your stated definitions
2. Computes logical implications
3. Generates inferred relationships

Why it matters:

- Ensures logical consistency
- Finds relationships you didn't state explicitly
- Helps catch modeling errors (concepts landing in wrong places)

Workflow:

1. Author content (stated view)
2. Run classification
3. Review results (inferred view)
4. Refine if needed

7.5. Decision Guide

Use Stated When:

- Creating/editing concepts
- Understanding what was explicitly modeled
- Debugging classification issues

Use Inferred When:

- Browsing for end users
- Searching for concepts
- Validating classification results
- Determining complete logical relationships

8. Navigation Coordinates

8.1. Overview

Navigation coordinates answer: **"How do I browse hierarchies?"**

They control which hierarchies to display, whether to show inactive concepts, and how to sort children.

8.2. The Three Components

8.2.1. 1. Navigation Patterns

Which hierarchy to use:

- **Inferred IS-A** - Complete logical hierarchy (most common)

- **Stated IS-A** - Authored relationships only
- **Custom** - Organization-specific (part-of, location-of, etc.)

8.2.2. 2. Vertex States

Which concepts to include:

- **Active only** - Current concepts (most common)
- **Both** - Active and inactive (for authoring, QA)
- **Inactive only** - Retired concepts (rare)

8.2.3. 3. Sorting

How to order children:

- **Alphabetical** - Easy to find (most common)
- **Natural order** - Unsorted, preserves creation order
- **Custom** - Domain-specific (by frequency, severity, etc.)

8.3. Stated vs. Inferred Hierarchies

Stated: Sparse, shows only what you authored

- Use for: Authoring, understanding modeling

Inferred: Complete, shows all logical relationships

- Use for: Browsing, searching, end users

8.4. Common Configurations

Use Case	Navigation Settings
End user browsing	Inferred IS-A, Active only, Alphabetical
Content authoring	Stated IS-A, Both active/inactive, Natural order
QA review	Inferred IS-A, Both, Alphabetical
Custom workflow	Custom pattern, Active only, Custom sort

8.5. Decision Guide

Pattern:

- Inferred IS-A → General browsing, searching
- Stated IS-A → Authoring, debugging
- Custom → Specialized workflows

Vertex States:

- Active only → End users
- Both → Authoring, QA

Sorting:

- Alphabetical → General use
- Natural order → Development
- Custom → Specialized workflows

9. Edit Coordinates

9.1. Overview

Edit coordinates answer: **"Who am I, and where does my work go?"**

They control authoring attribution and content organization.

9.2. The Five Components

9.2.1. 1. Author for Changes

Your identity: Individual name, role, team, or system

Why: Accountability, quality tracking, audit trails

9.2.2. 2. Default Module

Where new content is created:

- Core module
- National extension
- Local extension

Note: Applies to new content only. Existing content stays in original module.

9.2.3. 3. Destination Module

Where content moves during modularization (reorganizing between modules)

9.2.4. 4. Default Path

Which branch to write to:

- Development - Active authoring

- Staging - Testing
- Production - Released (usually read-only)

9.2.5. 5. Promotion Path

Target branch for promoting content (Development → Staging → Production)

9.3. The Three Workflows

9.3.1. 1. Developing

Creating or modifying content:

- New concepts → Default module, development path
- Existing concepts → Original module preserved, new version on development path

9.3.2. 2. Modularizing

Moving content between modules:

- Selected concepts → Destination module
- Creates new versions with module change

9.3.3. 3. Promoting

Moving content between paths:

- Copies from source path to promotion path
- Typical: Development → Staging → Production

9.4. Standard Development Pipeline

Development Path → Staging Path → Production Path

- **Development**: Author and refine
- **Staging**: QA and validation
- **Production**: Release to end users

9.5. Best Practices

- Verify edit coordinate before authoring
- Use project modules for temporary work
- Never author directly on Production
- Always promote through staging for QA

9.6. Common Configurations

Role	Edit Settings
Terminology Editor	Author: Your name, Module: Local Extension, Path: Development
QA Reviewer	Author: QA identity, Path: Staging, Promotion: Production
Import Process	Author: System process, Module: Core, Path: Integration branch

10. Practical Scenarios

10.1. Scenario 1: Clinical Terminology Browser

Context: Hospital terminology browser for clinicians

Requirements: Active concepts, US English, current content, production-quality

Coordinate Configuration:

- STAMP: Active only, Latest, Production
- Language: US English preferred terms
- Logic: Inferred relationships
- Navigation: Inferred IS-A, Active only, Alphabetical
- Edit: Not applicable (read-only)

Result: Clinicians see complete, current, active concepts with readable names.

10.2. Scenario 2: Content Author Workstation

Context: Knowledge engineer creating local hospital extension concepts

Requirements: See all content, work on development, validate with classification

Authoring Configuration:

- STAMP: Both active/inactive, Latest, Development
- Language: English, fully qualified names
- Logic: Stated relationships
- Navigation: Stated IS-A, Both, Natural order
- Edit: Author name, Local Extension module, Development path

Validation Configuration (after classification):

- Logic: Inferred relationships
- Navigation: Inferred IS-A, Alphabetical

Workflow:

1. Author with stated view
2. Run classification
3. Switch to inferred view to validate
4. Refine if needed
5. Promote to Staging when ready

10.3. Scenario 3: Multilingual Patient Portal

Context: Patient-facing health portal serving English and Spanish speakers

Requirements: Prefer Spanish, fallback to English, patient-friendly terms

Coordinate Configuration:

- STAMP: Active only, Latest, Production
- Language: Spanish preferred terms → English fallback
- Logic: Inferred relationships
- Navigation: Custom patient hierarchy, Active only, Custom sort
- Edit: Not applicable (read-only)

Result: Patients see content in preferred language with graceful fallback and patient-appropriate terminology.

10.4. Key Configuration Patterns

Read-Only End User:

- Active only, Latest, Production
- Preferred terms in user's language
- Inferred hierarchy, Alphabetical

Content Authoring:

- Both active/inactive, Latest, Development
- Stated (authoring) → Inferred (validation)
- Author identity, Extension module

QA Review:

- Both active/inactive, Latest, Staging
- Inferred hierarchy
- QA reviewer identity, Promotion to Production

11. Best Practices

11.1. Top 10 Coordinate Practices

11.1.1. 1. Know Your Role

Set coordinates appropriate for your role:

- Editor: Development path, stated view
- QA: Staging path, inferred view
- End User: Production path, active only

11.1.2. 2. Verify Before Authoring

At session start, confirm:

- Author: Your identity
- Module: Correct project module
- Path: Development (NOT production!)

11.1.3. 3. Default to Active Only for End Users

Clinical and business users should see active content only. Both active/inactive is for authors and QA.

11.1.4. 4. Work on Development, Release from Production

Never author directly on Production. Always: Development → Staging → Production.

11.1.5. 5. Use Cascading for Multilingual

Always provide fallback languages:

- Primary language
- Fallback language (usually English)
- Last resort: Any language

11.1.6. 6. Author with Stated, Browse with Inferred

- Author: Stated hierarchy (see what you're modeling)
- Validate: Inferred hierarchy (see classification results)
- End Users: Inferred hierarchy (complete view)

11.1.7. 7. Match Description Types to Audience

- Clinicians: Preferred Terms
- Patients: Synonyms or Preferred Terms
- Technical Staff: Fully Qualified Names
- Help Systems: Definitions

11.1.8. 8. Classify Regularly During Development

Don’t wait weeks between classifications. Classify daily during active development to catch issues early.

11.1.9. 9. Keep Coordinates Consistent During Work Sessions

Don’t constantly switch coordinates. Set them at session start and keep them stable for coherent work.

11.1.10. 10. Share Coordinate Context When Collaborating

When discussing content with teammates, specify which coordinates you’re using. "I don’t see that!" often means different coordinates.

11.2. Common Pitfalls to Avoid

Pitfall 1: Forgetting You’re on Development Path

Users can’t see your work because you’re viewing Development, they’re viewing Production.

Solution: Be aware of your path. Promote when ready.

Pitfall 2: Using Stated Hierarchy for End Users

Stated is sparse and incomplete for browsing.

Solution: End users need inferred hierarchy.

Pitfall 3: Showing Inactive Concepts to End Users

Confuses users with retired content.

Solution: Active only for clinical interfaces.

Pitfall 4: Not Cascading Languages

Blank descriptions when translation missing.

Solution: Always use fallback languages.

Pitfall 5: Authoring on Production

Untested content reaches users immediately.

Solution: Author on Development only.

11.3. Quick Checklist

Before Authoring:

- ▢ Author identity correct?
- ▢ Default module correct?
- ▢ Default path = Development?
- ▢ Status = Both (to avoid duplicating retired concepts)?

Before Promoting to Production:

- ▢ Classification complete?
- ▢ QA review passed?
- ▢ Stakeholder approval?
- ▢ Release notes prepared?

For End User Interfaces:

- ▢ Status = Active only?
- ▢ Path = Production?
- ▢ Language appropriate?
- ▢ Navigation = Inferred hierarchy?

11.4. Summary

Coordinates are powerful filters that control what you see and how you interact with terminology. Use them skillfully:

- Set them appropriately for your role
- Keep them consistent during work
- Verify before critical operations
- Understand common patterns
- Avoid common pitfalls

Master coordinates, and you'll be a more effective knowledge engineer.

12. Quick Reference

Common Configurations:

Use Case	Key Settings
Clinical Browsing	Active only, Latest, Production, Inferred hierarchy
Content Authoring	Both active/inactive, Development, Stated hierarchy
QA Review	Both active/inactive, Staging, Inferred hierarchy
End Users	Active only, Latest, Production, Preferred terms

13. Glossary

Active
: Current, usable content

Author
: Person or system who created/modified content

Inferred
: Relationships computed by classifier

Module
: Organizational grouping of content

Path
: Development branch (Development, Staging, Production)

STAMP
: Status, Time, Author, Module, Path

Stated
: Relationships explicitly authored

Version
: Specific state of content at a point in time