GAPs Analysis & Ontology Mapping

Reference Document: `Apps/mygit/a4b-doc-xtalent/product/xTalent/docs/GAPs.xlsx`
Sheet: Mapping BA vs UI
Focus: Column C (Concepts) & D (Attributes), specifically Yellow Highlighted items.

1. Overview

This document provides a detailed mapping and explanation for the GAP items identified in the BA analysis. It connects the business concepts and terminology found in the GAPs document to the standardized Ontology and Concepts defined in the xTalent Core, Time & Absence, and Total Rewards modules.

The primary goal is to resolve ambiguity around the "Yellow Highlighted" concepts and demonstrate how the v2.0 Architecture supports these requirements.

2. Yellow Highlighted Items Analysis (Column C)

The following 15 items were highlighted in the source document. Below is the mapping to the xTalent Ontology.

#	BA Concept (Col C)	Target Ontology Entity	Explanation / Resolution
1	**Company**	`LegalEntity`	**Direct Mapping**. In the Core module, a formal company legal structure is represented by `LegalEntity`. It supports hierarchy (Groups, Subsidiaries) and country-specific compliance settings.
2	**Unit HR**	`BusinessUnit` (aka Unit)	**Mapped to Attribute/Type**. "Unit HR" refers to an organizational unit that serves an HR function. In the ontology, this is a `BusinessUnit` (specifically

#	BA Concept (Col C)	Target Ontology Entity	Explanation / Resolution
			`Unit` entity) with a `UnitType` of `DEPARTMENT` or `TEAM`. It can be tagged via `UnitTag` as an HR function or identified by its Department code. If it represents a reporting line, it maps to `Supervisory Organization`.
3	**Area Type**	`AdminArea` (Metadata/Level)	**Mapped to Attribute**. Geography is handled by `Country` and `AdminArea`. "Area Type" corresponds to the `level` (1=Province, 2=District, 3=Ward) or `metadata.type` (e.g., "PROVINCE", "CITY") of an `AdminArea`.
4	**Track**	`JobTaxonomy`	**Mapped to Taxonomy**. "Track" (e.g., Technical Track, Management Track) is part of the `JobTaxonomy` tree structure. It can be a node in the taxonomy or defined in `JobProfile` metadata as `career_track`.
5	**Job Family**	`JobTaxonomy` (Level 1)	**Direct Mapping**. `JobFamily` is explicitly supported as Level 1 in the `JobTaxonomy` tree. It groups jobs by broad occupational categories (e.g., Engineering, Sales).
6	**Job Group**	`JobTaxonomy` (Level 2)	**Direct Mapping**. `JobGroup` fits as Level 2 in the `JobTaxonomy` hierarchy, falling under Job Family. The multi-level tree supports arbitrary depth if needed.
7	**Sub Group**	`JobTaxonomy` (Level 3+)	**Direct Mapping**. `Sub Group` is a child node of `JobGroup` within the `JobTaxonomy` tree.

#	BA Concept (Col C)	Target Ontology Entity	Explanation / Resolution
8	**Job**	`Job`	**Direct Mapping**. The `Job` entity defines the standardized role (e.g., "Senior Software Engineer") independent of the person holding it. It links to Family, Grade, and Level.
9	**Ranking Level (Job Level)**	`JobLevel`	**Direct Mapping**. Matches `JobLevel` entity which defines career progression tiers (e.g., Junior, Senior, Principal). It uses `level_order` for ranking.
10	**Document Status**	`Document` / `Contract` Status	**Mapped to Attribute**. Status fields exists on all major entities. For contracts, it is `Contract.status_code` (Draft, Active, Expired). For general documents, it maps to the document management system's status metadata.
11	**Contract Type Group**	`CodeList` / Metadata	**Mapped to Grouping**. `ContractType` is a standard code (e.g., Labor Contract, Probation). "Group" is a higher-level classification (e.g., "Indefinite" vs "Definite") managed via `CodeList` metadata or parent codes.
12	**Contract Type**	`Contract.contract_type_code`	**Direct Mapping**. Defined in `glossary-employment.md`. Values include `PERMANENT`, `FIXED_TERM`, `PROBATION`, etc.
13	**Appendix Contract Type**	`Contract` (Child)	**Mapped to Relation**. An Appendix is a `Contract` entity that links to a `parent_contract_id`. The "Type" distinguishes it as an Amendment or

#	BA Concept (Col C)	Target Ontology Entity	Explanation / Resolution
			Appendix via `contract_type_code` or metadata.
14	**Change Type Group**	`ReasonGroup`	**Mapped to Configuration**. Grouping of change reasons (e.g., "Compensation Changes", "Job Changes") used in UI and workflow rules. Managed via system configuration/CodeList.
15	**Change Type**	`ReasonCode`	**Direct Mapping**. When an assignment or record changes, a `ReasonCode` is required (e.g., `PROMOTION` , `TRANSFER` , `MERIT_INCREASE`). This tracks *why* a change occurred in `EmploymentHistory` .

3. General Concepts (Columns C & D)

Beyond the highlighted items, the document lists other concepts which map as follows:

- **Office**: Maps to `WorkLocation` or `Facility` .
- **Market**: Maps to `GeographicRegion` or `TalentMarket` (see `glossary-talent-market.md`).
- **Hierarchical Chart**: Generated view based on `BusinessUnit` (parent/child) and `Assignment` (manager/subordinate) relationships.
- **Organization Class**: Maps to `UnitType` (Division, Department, etc.).
- **Job Grade**: Maps to `JobGrade` (Pay Grade).
- **Career Track**: Maps to `JobTaxonomy` or separate `CareerTrack` configuration in Career Development module.
- **Currency**: Standard ISO currency codes used in `JobGrade` and `Compensation` .
- **Ethnicity / Religion / Marital Status**: Standard demographic fields on `Person` entity (`glossary-person.md`).

5. Detailed Attribute Mapping

This section details how specific legacy attributes map to the v2.0 Entity-Attribute model.

5.1. Company (→ `LegalEntity` + Profiles)

Legacy "Company" aggregates legal, profile, and banking data. In v2.0, this is normalized across related entities.

Legacy Field	v2.0 Target Entity	v2.0 Attribute	Notes
Code	`LegalEntity`	`code`	Unique identifier
Name	`LegalEntity`	`name_vi / name_en`	Local/Global names
Type	`EntityType`	`code`	e.g., HOLDING, COMPANY, BRANCH
Parent Company	`LegalEntity`	`parent_id`	Hierarchy link
Company Group	`LegalEntity`	`metadata.group`	Or implied by hierarchy root
Address 1	`EntityProfile`	`address1_street`	
Phone, Fax, Email	`EntityProfile`	`phone` , `fax` , `email`	Profile contact info
Website, Tagline	`EntityProfile`	`website` , `tagline`	Branding info
CEO	`EntityProfile`	`ceo_worker_id`	Ref to `Worker`
Tax ID	`EntityProfile` (or `EntityLicense`)	`tax_id`	Tax registration
Bank Account	`EntityBankAccount`	`account_number`	1-to-many relation

Legacy Field	v2.0 Target Entity	v2.0 Attribute	Notes
Business License	`EntityLicense`	`license_number`	1-to-many relation
Legal Representative	`EntityRepresentative`	`worker_id`	1-to-many relation
File đính kèm	`EntityLicense`	`metadata.attachment_url`	Attachments link to documents

5.2. Unit HR (→ `BusinessUnit`)

"Unit HR" is a functional department.

Legacy Field	v2.0 Target Entity	v2.0 Attribute	Notes
Code	`BusinessUnit`	`code`	
Name	`BusinessUnit`	`name`	
Parent Department	`BusinessUnit`	`parent_id`	Operational hierarchy
Organization Class	`UnitType`	`code`	DIVISION, DEPARTMENT, TEAM
Manager	`BusinessUnit`	`manager_employee_id`	Direct unit manager
Cost Center	`BusinessUnit`	`cost_center_code`	Financial dimension
Effective Date	`BusinessUnit`	`effective_start_date`	SCD Type 2 tracking
HRBP/RAMs Group	`RelationEdge`	-	Matrix relationship (Unit → HRBP Worker)
HRBP Code/Name	`Worker` (via Relation)	`code` , `name`	The person assigned as HRBP

5.3. Contract Type (→ `CodeList`)

Contract types are reference data stored in `CodeList` with rich metadata.

Legacy Field	v2.0 Target Entity	v2.0 Attribute	Notes
Code	`CodeList`	`code`	Group: `CONTRACT_TYPE`
Name	`CodeList`	`display_en`	
Contract Type Group	`CodeList`	`metadata.group_category`	e.g. "Definite", "Indefinite"
Employee Type	`CodeList`	`metadata.allowed_employee_type`	Filter for UI
Duration (Days)	`CodeList`	`metadata.duration_days`	Default duration
Duration Name	`CodeList`	`metadata.duration_label`	Display label
Active	`CodeList`	`is_active`	

5.4. Appendix Contract Type (→ `CodeList`)

Similar to Contract Type but for amendments.

Legacy Field	v2.0 Target Entity	v2.0 Attribute	Notes
Code	`CodeList`	`code`	Group: `APPENDIX_TYPE`
Is Extension	`CodeList`	`metadata.is_extension`	Flag
Is Change Salary	`CodeList`	`metadata.is_change_salary`	Flag
Notice Period	`CodeList`	`metadata.notice_period_days`	

5.5. Change Type (→ `EventReason` / Configuration)

"Change Type" defines the rules for personnel actions (workflow, forms, fields). This maps to a configuration layer for **Events**.
Current v2.0 Core Ontology stores the list of reasons in `CodeList` (Group: `ASSIGNMENT_REASON`), but

*the complex configuration (Form Type, Fields) belongs to the **Business Rules / Process Configuration** layer (potentially `Workflow` module).*

Legacy Field	v2.0 Target Entity	v2.0 Attribute	Notes
Code	`CodeList` (Reason)	`code`	Group: `EVENT_REASON`
Name	`CodeList` (Reason)	`name`	e.g. "Promotion", "Transfer"
Signer Title	`WorkflowConfig`	`approver_role`	Approver logic
Change Form Type	`CodeList` (Reason)	`metadata.form_template_id`	UI Form to load
Show in Working Record	`CodeList` (Reason)	`metadata.history_visibility`	Visibility flags
Field Manage	`DynamicFormConfig`	`fields_config`	Defines visible/required fields for this action

4. Resolution Strategy

The xTalent v2.0 Ontology coverage is sufficient to handle these gaps.

- **Action**: No new entities are needed.
- **Configuration**: The mapping requires configuring `JobTaxonomy` trees, `UnitTypes`, and `CodeLists` (for types/reasons) during implementation.
- **Migration**: Data from the legacy system (matches in GAPs.xlsx) maps cleanly to these v2.0 structures.