BUSINESS DOMAIN MODEL

VISUAL MODEL OVERVIEW

A Business Domain Model is a conceptual view of all or part of an enterprise focusing on products, deliverables, and events that are important to the mission of the organization. A Business Domain Model logically represents the business concepts to be implemented by the system, the relationships between them, and identifies their key attributes.

This Business Domain Model was created to identify the key content-related concepts already implemented by the system and show how new concepts being discussed could be incorporated into the overall business model.

***BABOK® Guide* Connection**

Technique 10.15
Data Modelling

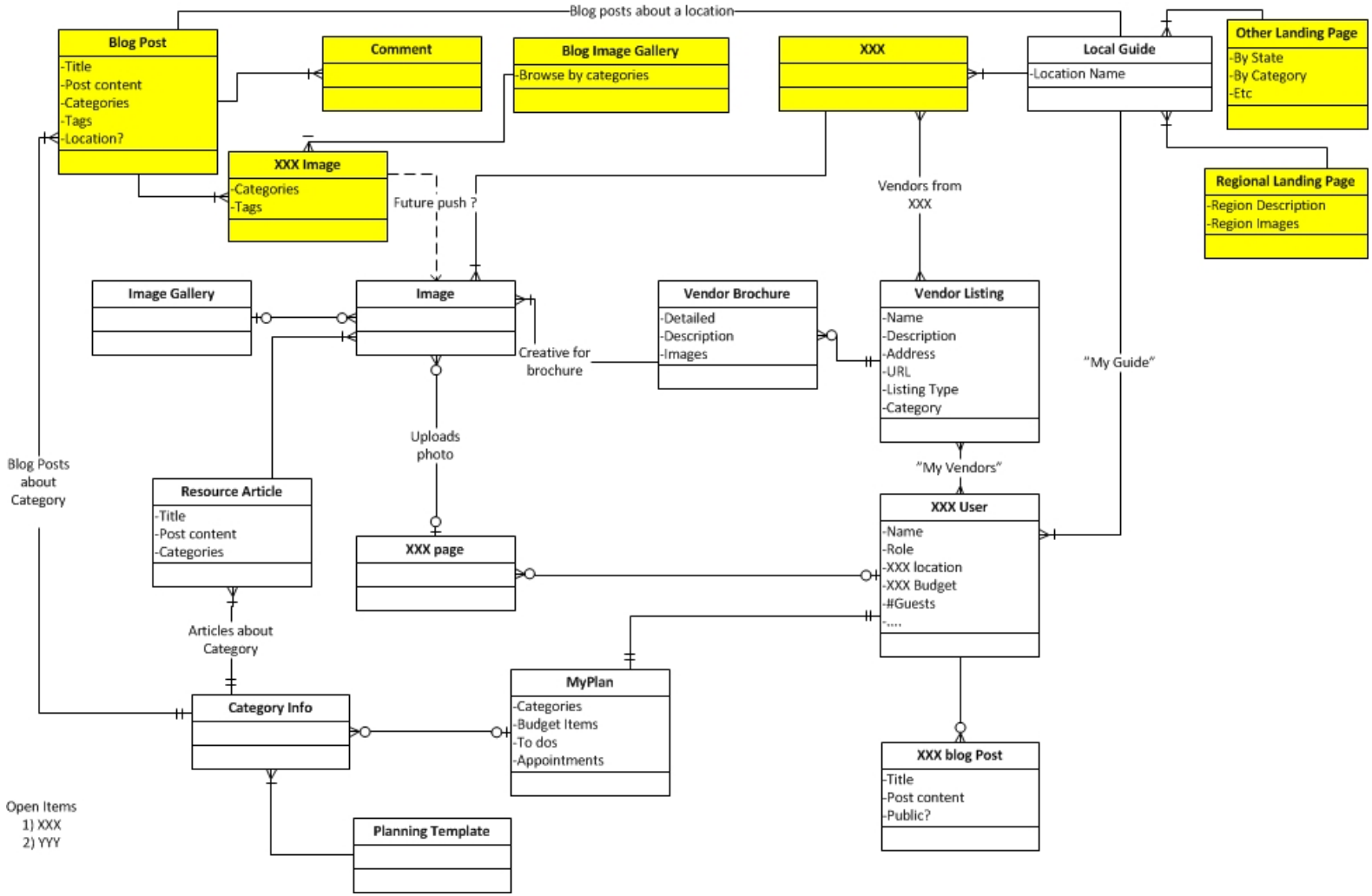
To describe the entities, classes, or data objects relevant to a domain, the attributes that are used to describe them, and the relationships among them.

The organization was quickly adding multiple different types of content and integrating new systems and sub-systems to manage content. There was a desire for end users to eventually be able to navigate seamlessly from one content type to another using the organization's web application. There was a concern that we were going to create a disconnected mess of content.

This model was presented as a stand-alone deliverable, attached in an email to the executive who requested it with a detailed explanation. As a result of this analysis, I was able to see that there were adequate connections between the different content types and alleviate the short-term concerns about creating a mess that couldn't easily be cleaned up later. Attention quickly turned to more pressing projects.

THE MODEL

Content Inventory



Detailed Description

Entities and Attributes

Each box represents a business concept in business terms. The business concept is named in bold within the box. The attributes or elements of the business concept are listed in each box. In this scenario, this model represented an early draft and the list of attributes is often incomplete.

Yellow highlighting of the boxes was used to represent new or soon-to-be-developed concepts.

Relationships

The lines connecting the boxes represent relationships between business concepts.

The ends of the lines represent different types of relationships:

Cardinality	Description
	Any number from one to many
	Zero to one
	Only one
	Any number (zero to many)

These notations are part of Crow's Foot Notation. Unified Modeling Language (UML) provides an alternative notation for modeling multiplicity.

Let's look at how this works. For example, the concept for Vendor Listing is connected to Vendor Brochure. A Vendor Listing can have any number of Vendor Brochures – it may be zero if the brochure is not created yet, or it may be one or more. A Vendor Brochure is linked to one and only one Vendor Listing.

You'll notice a few of these connections are listed with no cardinality. When creating this model I had incomplete information. I was also new to using the Crow's Foot Notation and overlooked the need to have cardinality at the end of

every line. I prefer the UML multiplicity notation as I find it simpler, however it wasn't supported by the software I had access to at the time.

TERMINOLOGY

A **Conceptual Data Model** is the term defined in the *BABOK® Guide* to refer to this type of model. Other comment terms include:

- **Class Diagram** – This term is used when the model supports object-oriented development. It is an alternative to an ERD.
- **Data Model (Logical)** – An alternate term is used for models like this one that represent high-level business concepts and not necessarily the implemented database model.
- **Entity Relationship Diagram (ERD)** is the formal term used to refer to the type of diagram in this example. ERDs can be used to model data, relationships, and attributes at multiple levels of specificity, all the way down to the physical database model.
- **Relationship Map** – A business model that shows the organizational context in terms of the relationship that exist among the organization, external customers, and providers.

POSSIBLE USES

Use this type of model:

- When there is lack of clarity about what the key business concepts are and/or how key business concepts relate.
- When the data model is tied to an old business model and you need to break discussions from the actual data structures to focus on business terminology.
- When you need to identify how upcoming changes modify or extend your business model.
- When you need to show how concepts and data are connected across system boundaries.

- When there is conflict regarding how to create or modify a data model to meet a set of requirements. A business domain model helps bring the discussion back to business needs, while still focusing on the relationships between different types of information, and can often bring clarity to technical design decisions.

How to Create a Similar Model

Follow these steps to create a similar model for your project.

1. Create a list of your key business concepts.
2. Draw connections between key concepts. (This technique is best done on paper, white board, or using index cards first so you don't drive yourself crazy arranging and re-arranging boxes on your screen as you discover new relationships.)
3. List the important attributes for each concept. If you find yourself wanting to nest attributes or group them together or link an attribute to a concept, that's a good sign that the attribute should be broken out as a concept instead.
4. Review your model to be sure that the concepts are generally at the same level of abstraction.
5. Conduct one or more walk-throughs with business and technical stakeholders to confirm your model.

Watch Out For

The most difficult aspect of creating a domain model is to break away from database concepts and terminology. This is especially difficult for technical stakeholders or any business users who are familiar with the data model. If you focus on the implemented model, you risk losing sight of important business concepts or making assumptions about business concepts. Encourage everyone to focus on business terminology and let go of database terminology for the purposes of this discussion. Data models can come later.