The Enlightened, Post-Modern Product Model

Maxim Noah Khailo & Jeff Mataya

February 12, 2016

1 Purpose

There is both a universal objective reality and our subjective interpretations of it. The post-modern world is one of flux and uncertainty which can only collapse into the shadow of objectivity when illuminated.

Below is a description of a product model, nay, a content model that both provides variants in the face of contextualization, but has its objective reality and definition.

Please read the companion paper "Everything is Search" after this one to realize the full potential of the approach described below.
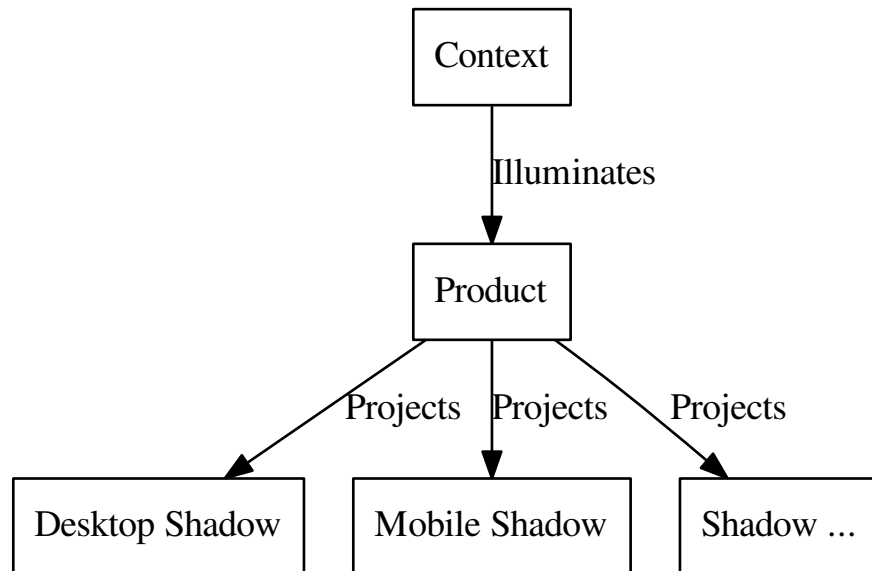
2 Forces

We have to balance several problems.

- Provide the most appropriate information at the right time.

- Semantically structure information into a computable form.

- Provide a manageable way to handle variants.

3 The Form and Shadow

The model is split into two concepts, the *Form* and the *Shadow*. A product takes on one form but can have many shadows. Shadows are created when a *Context* illuminates the *Form*.

```
                    ┌───────────┐
                    │  Context  │
                    └───────────┘
                          │
                          │ Illuminates
                          ▼
                    ┌───────────┐
                    │  Product  │
                    └───────────┘
              Projects │ Projects │ Projects
          ┌────────────┘         └────────────┐
          ▼            ▼                       ▼
  ┌────────────────┐ ┌────────────────┐ ┌────────────────┐
  │ Desktop Shadow │ │  Mobile Shadow │ │   Shadow ...   │
  └────────────────┘ └────────────────┘ └────────────────┘
```

3.1 Product

The product is a combination of schema and data. The model is composed
of dynamic attributes which have a type. Each attribute is a map of values
that have names. The Form may look like this, represented as JSON.

```
//Product
{
    id: 3
    name: {
        type: "string":
        x: "Red Shoe",
        y: "Big Red Shoe",
    },
    description: {
        type: "string"
        x: "Buy Me",
        y: "Purchase Me",
    },
    image: {
        type: "uri"
        x: "http://a",
        y: "http://b",
        z: "http://c",
    }
}
```

3.2   The Shadow

The Product Shadow is what is used when displaying a product to a customer. At any point in time, the customer has a Context. The Product Shadow is the shape you get when you illuminate the Form from a particular vantage point called the Context. Here is an example of a Shadow represented as JSON.

```
//Product Shadow for English Desktop
{
    context: { modality: "desktop", language: "english" },
    product: 3,
    name: "x",
    description: "x",
    image: "x",
}
```

And here is another, but for mobile...

```
//Product Shadow for English Mobile
{
    context: { modality: "mobile", language: "english" },
    product: 3,
    name: "y",
    description: "y",
    image: "z",
}
```

Each attribute in the Shadow corresponds to an attribute in the Form. It selects the value from the Form via it's name. For example, the 'x' value from the 'name' attribute in the Form..

4   The Context

Each *Shadow* is a representation of the product *Form* given a *Context*. Each user will have a Context when they interact with our system. The Context is used to select which shadows to display for a particular product.

If we assign statistics to each of the Shadow's attributes, as described in the companion paper "Everything is Search", this model will provide a solid foundation for algorithms that could modify shadows or create new ones based on data and statistics.

5   Mapping Variants to SKUs

The product varies in two different ways. It varies based on the context the user is in and it varies based on properties like size and color. Remember a context describes the environment the customer is in like "mobile and english"

Lets call the way the product varies by context the "Context Variant" and the way the product varies by property the "Property Varaint"

How do we map all the ways the product varies to a SKUs?

The Product can have "Context Variants" which represent trees where the leaf nodes are the SKUs. A walk down the tree represents a unique ID. Given the product ID and this walk down the tree we get a mapping of product to SKU.

```
{
    context1 : {
        color : {
            red : "SKU–RED1" ,
            green : "SKU–GREEN2" ,
        }
    }
    context2 : {
        color : {
            purple : "SKU–BLUE3" ,
            orange : "SKU–ORGAN3" ,
        }
    }
    ...
}
```

The benefit of this model is we can represent arbitrarily complex variants by simply assigning the leaves of the tree with SKU values. If the leaf nodes are empty, we can have an algorithm that can walk the tree and assign meaningful SKU numbers.

Also, because of our Shadow model, we can handle cases such as "Only purple and orange are available in Germany." if the user is in a German context.

Image a product has the following variants

```
//Product With Variants
{
    Germany : {
```

```
        color : {
            red : "SKU–RED1",
            green : "SKU–GREEN2",
        }
    }
    USA : {
        color : {
            purple : "SKU–BLUE3",
            orange : "SKU–ORGAN3",
        }
    }
    ...
}
```

We can map the variant in the product form and shadow like so.

```
//Product with Variant Attributes
{
    id: 3
    name: {
        type: "string":
        USA : {
            color : {
                purple: "Purple Shoe",
                orange: "Sun Orange Shoe",
            }
        }
        German : { ... }
    }
    ...
}

//Product Shadow Using "USA" variant
{
    context: { language: "english", region: "USA" }
    product: 3,
    name: "usa",
    description: "english"
}
```

The benefit of this approach is that not all attributes have to change based on variants But it allows some to change. For example, the description

attribute above doesn't change based on variant, but the name attribute does.

Also we can reuse different variants and attributes in different contexts.

6   SKUs

The combination of Product and Product Shadow gives us a way to represent the way that a Product Details Page (PDP) is represented to the user. For that reason, these data models are fundamentally used for acts of merchandising.

However, in many cases the basic attributes of the SKU will change based on the context. Price, for example, has this behavior. Taking a simple example will show this behavior.

Foxy Suncare is selling sunscreen that comes in three sizes: trial (4oz), regular (8oz), and jumbo (16oz) at costs of $8.00, $15.00, and $22.00. Since all of these items would appear on a single PDP, they are all represented by the same Product, and same set of Product Shadows. Naturally, we would like to see price then be an attribute of SKU.

And since price may change based on context (for example, the sunscreen myght be sold at different prices in the United States vs Thailand), we need to apply the Shadow concept to SKUs.

```
// SKU
{
    id: 1,
    sku: "SUNSCREEN-TRAVEL",
    description: {
        "USA": "SPF-5,_so_you_can_feel_like_you're_wearing_protection_
    },
    price: {
        "USA": 800,
        "THAILAND": 2300,
        "UK": 650
    }
}

// SKU Shadow for USA
{
    id: 1,
    description: "USA",
    price: "USA"
}

// SKU Shadow for Thailand
{
```

```
    id : 1 ,
    description : "USA" ,
    price : "Thailand"
}
```

7   Models

7.1   Context

A Context has an identifier, name, and attributes.

```
case class Context (
    id : Id ,
    name : String ,
    attributes : Json )
```

Where the attributes can be arbitrarily specific, but should be of a flat key/value nature.

```
{
    modality : "x" ,
    language : "y"
}
```

7.2   Product

The Productrepresents a product as described above.

```
case class Product (
    id : Id ,
    attributes : Json ,
    variants : Json )
```

The attributes should be of the form

```
{
    attribute1 : {
        type : "abc" ,
        val1  : "x" ,
        val2  : "y" ,
        . . .
    }
    attribute2 : {
```

```
            . . .
        }
    }
```

7.3   ProductShadow

The ProductShadow represents a Shadow of a product as described above.

```
case class ProductShadow(
    id: Id,
    context: Id,
    productForm: Id,
    attributes: Json)
```

Where the attributes correspond to attributes in the product form, selected by name.

```
{
    attribute1: "x",
    attribute2: "...",
    . . .
}
```

7.4   Sku

The Sku represents a product as described above.

```
case class Sku(
    id: Id,
    sku: String,
    attributes: Json)
```

The attributes should be of the form

```
{
    attribute1: {
        type: "abc",
        val1 : "x",
        val2 : "y",
        . . .
    }
    attribute2: {
        . . .
```

```
        }
    }
```

7.5   SkuShadow

The SkuShadow represents a Shadow of a SKU as described above.

```
case class SkuShadow(
    id: Id,
    context: Id,
    sku: Id,
    attributes: Json)
```

Where the attributes correspond to attributes in the Sku, selected by name.

```
{
    attribute1: "x",
    attribute2: "...",
    ...
}
```

7.6 Tiny MVP

For Tiny MVP we will seed only 1 default context. The product form will have one "default" variant and minimal set of attributes.

```
//MVP Product
{
    title: {
        type: "string",
        ...
    }
    description: {
        type: "string",
        ...
    }
    images: {
        type: "images"
        ...
    }
    price: {
        type: "price"
        ...
    }
}
```

We provide the following Admin endpoints

- GET /api/v1/contexts

- GET /api/v1/products/x

- GET /api/v1/product-shadows/x

- POST /api/v1/product-shadows/

- PUT /api/v1/product-shadows/x

- GET /api/v1/product-forms/x

- POST /api/v1/product-forms/

- PUT /api/v1/product-forms/x

- GET /api/v1/skus/x

- POST /api/v1/skus/

- PUT /api/v1/skus/x

- GET /api/v1/sku-shadows/x

- POST /api/v1/sku-shadows/

- PUT /api/v1/sku-shadows/x

The "/api/v1/products/" route will implement what eventually the storefront JSON will look like.

We can index everything in ES using context like this.

- /api/search/products-[context]/

- /api/search/product-shadows-[context]/

- /api/search/product-forms-[context]/

- /api/search/skus-[context]/

- /api/search/sku-shadows-[context]/