Applying Semantic Web technologies in Product Information Management at NXP Semiconductors

Parvathy Meenakshy (parvathy.meenakshy@nxp.com),
John Walker (john.walker@semaku.com)
2014-09-21

Abstract

In the electronics industry, the ability to get accurate, timely product data in front of the customer is a key factor in the overall business process. In this paper we describe how NXP is making use of Semantic Web technology such as RDF and SPARQL to manage a product taxonomy for marketing purposes that forms the key navigation of the NXP website (http://www.nxp.com) and the next steps to extend this to create a domain model covering applications, technologies and other key entities that can be used to create rich user journeys through the content.

Introduction

The ease of finding the right product is paramount for the customer and crucial for sales, but at the same time a complex problem in an eCommerce scenario. Enabling the customer to easily find, compare and select the right product can reduce the overall time and therefore costs involved in the purchasing process. In the B2B electronic component industry this can involve the choice between literally thousands of candidates. We believe opening up access to the product data is a key enabler for this process. Whether to free the data from existing silos for use within the organization or to make the data available to third parties such as distributors and search engines.

In this paper we focus on the development and management of taxonomies and describe how NXP has deployed a solution based on SKOS for the management of the product taxonomy and a currently-under-development approach to provide an alternate view of the product catalog.

Product taxonomy

The NXP product taxonomy [1] provides a marketing-oriented categorization for the product catalog. The taxonomy consists of categories organized into a hierarchical structure used for navigation on the NXP website. The taxonomy is primarily function-oriented, but also incorporates market, application and technology based categorization of products. It is permitted for a category to have more than one parent, forming a polyhierarchy. The taxonomy is used to manually position products into one or more categories as required for marketing purposes. Additional content and documents can be linked to a category.

Legacy approach

The product taxonomy is managed in the enterprise Product Information Management (PIM) system where each category is assigned a numeric identifier. Placement of products in the taxonomy is also done in the PIM system. The taxonomy and product assignments are exported from the PIM system in a proprietary XML format and loaded into an XML database. The document management system is able to query the XML database using XQuery to lookup categories to which a document can be linked using the numeric identifier. The XML is also imported to the web content management system and used to generate the website. Due to end of life of the PIM system, it was necessary to find an alternative approach for the management of the taxonomy.

Linked data approach

Previously we had created a process to convert the existing XML to RDF/XML and load the result into a graph store where it can be merged and queried along with data from other sources. For this we had created a mapping to SKOS where each category has the type `skos:Concept` and the hierarchical relations are mapped to the `skos:narrower` and `skos:broader` properties.

A logical next step was to make the RDF as the source and manipulate the data directly in the graph store using SPARQL 1.1 Update. After considering various open-source and commercial tools we opted to use SKOSjs [2] with some minor NXP-specific customizations and configuration.

For placing products into the tree we developed a simple application using the Play Framework that allows a user to manage the links. The application makes use of stored queries in the graph store that can be exposed as an HTTP API. Initial bindings for variables in the query can be passed as parameters in the request URI.

To support existing applications the legacy XML format is generated by transforming an RDF/XML dump of the data in the store using XSLT.

The benefits of this approach are multiple. First and foremost we have minted URIs as globally unique identifiers for each product category that can be used to unambiguously refer to the resource. The flexibility of the RDF data model has allowed us to add additional information such as alternate and translated labels for categories without disrupting existing applications. Additionally being able to use the power of SPARQL to flexibly query the data as a graph has opened up new ways to analyze the data and do quality and consistency checks. Users have also responded positively to having several simpler and more focused editing tools as opposed to the complex, yet generic user interface of the previous PIM system. In future we expect to gain further benefits from this approach.

Solution selling

For NXP the customer is typically an electronics design engineer. A common user story is a user who searches online for solutions to their design challenges rather than for a specific named component. Therefore NXP desires to serve the customer better by developing a solution-oriented view of the portfolio orthogonal to the existing function-oriented product taxonomy.

The specification of a product usually includes an textual applications chapter that lists the end-equipments, market segment and solution area in which the component can be used. This content can be indexed by site search and only gives the users a basic keyword search functionality. It does not provide an option for customers to explore the content under different perspectives or filter the data under search facets. Lack of a unique identifier for these named entities restricts reuse and aids ambiguity in referring to this information in different document assets.

A revamp of this static list of information would enable more effective and efficient discovery and selection of components which fit to a particular application/solution area and open up new opportunities for cross-selling related products and telling compelling stories about a particular focus area by aggregating relevant content. It should be noted that the application information of products does not always fit into a strict hierarchical structure. Moreover this information is subjected to change due to constant flux in technology trends and introduction of new products which fit into latest applications. A technology which is flexible enough to cope with the changes in schema is of prominence in our case. Semantic Web and Linked Data technologies enable an incremental development of domain model contrary to the conventional relational database schema. By applying these to describe the knowledge domain we can resolve ambiguities and empower machines to access the information more easily.

As described above we have already built experience with converting the NXP product taxonomy to RDF using the SKOS vocabulary where it is now stored and managed natively as RDF in a graph store. Additionally we have developed a simple web application that allows products to be linked to the taxonomy. Based on this success we are extending our use of semantic technologies by developing a domain model for applications/solutions that will drive a rich user-focused experience. This was based on the lessons learned from using SKOS where the terms like broader and narrower are very generic, whereas we need to have more domain-specific terminology to better capture the exact meaning. The approach taken was to first analyze the existing application content to extract common terms followed by domain modeling undertaken with the relevant subject-matter experts. Based on this model we have made wireframe designs for the various pages about the 'things' in the model which are due to be implemented in the coming months. During development of the model we encountered difficulties to explain the benefits of domain-driven design over a "UX first" approach which can often deliver tangible results faster, but can give a brittle and inflexible design. Areas we wish to investigate further are automated tagging of products and content assets (documents, images, video) with the concepts from the model and the possibility for a curatorial/personalization layer.

The approach

The approach used is domain driven design rather than presentation and interaction of UX design. One of the challenges was to identify an entity as a concept, story about a concept or a web page about a concept. Another was the difficulty in defining the concept in a data element fashion. This exercise is not so intuitive as we are not dealing with concrete or tangible things. We tried to map instances of data to the model and came up with potential user journey to agree on the relations between concepts. While trying to describe the semantic relation of application information to products, the priority was to conform to existing standards. SKOS and Dublin Core standards are used to express the basic metadata. But the semantics of SKOS is not rich enough to capture the application information of our products. Even though Schema.org is a light weight standard, defining all the concepts in our model with 'Product' class was not desirable.

Based on the model, instances were added with unique identifier or URI and are persisted to a named graph in a graph store (http://dydra.com/). A browsable interface of this data is created using the Linked Data API specification, implemented by open source project (https://code.google.com/p/puelia-php/). The API layer is configured to support REST API calls to the graph store. This provides the results in the HTML page without writing complex SPARQL queries and in different output format like RDF/XML, JSON etc. The ability to browse the data as HTML enables us to validate the model with the relevant subject matter experts. Later we expect that the same API layer can be used during generation of the pages on the NXP website.

Possible future work

One of the areas where we need to research further is on the auto tagging methods. This is of interest as automating the linking of content assets to concepts would reduce the manual work involved.

Similarly we would like to automate the categorization of products into the taxonomy by somehow expressing rules or constraints for the 'type' of products that should appear in the category and use this to automatically populate the product lists. As large amounts of the product specification are already available as structured data this is certainly possible, the real challenge is to make the management of these rules usable for the target audience.

References

[1] NXP product taxonomy http://www.nxp.com/products/
[2] SKOSjs https://github.com/tkurz/skosjs