The Global Language of Business

GS1 SmartSearch Implementation Guideline

Create web pages that refer to trade items (formerly GTIN+ on the Web)

Release 1.0.1, Ratified, Nov 2015

1 Document Summary

Document Item	Current Value
Document Name	GS1 SmartSearch Implementation Guideline
Document Date	Nov 2015
Document Version	1.0
Document Issue	1
Document Status	Ratified
Document Description	Create web pages that refer to trade items (formerly GTIN+ on the Web)

2 Contributors

Name	Organisation
Richard McKeating, working group co-chair	Tesco Stores
Mark Frey, working group facilitator	GS1 Global Office
Mark Harrison, document editor	Auto-ID Labs Cambridge
Ken Traub, document editor	Ken Traub Consulting LLC
Anarkat, Dipan	GS1 Global Office
Ausili, Andrea	GS1 Italy
Beideman, Robert	GS1 Community Room Staff
Benhaim, Marc	GS1 France
Bos, Frits	GS1 Netherlands
Bratt, Steve	GS1 GO
Burd, Randy	MultiAd Kwikee
Cornelisse, Rutger	Business Technologies Ltd
Cox, Christie	Wal-Mart Stores, Inc.
Dean, Kevin	GS1 Canada
Farance, Frank	Farance Inc.
Feuerstein, Véra	Nestle
Fuchs, Klaus	Auto-Id Labs ETH Zurich
Fuessler, Andreas	GS1 Germany
Gerasimenko, Alexander	Mars, Inc.
Gomez Sepulveda, Juan Pablo	GS1 Mexico
Gormley, Alan	GS1 Ireland
Hakala, Pertti	GS1 Finland
Hogan, Bernie	GS1 US
Houlette, Cedric	GS1 France
Kauz, Eric	GS1 Global Office
Kravec, Nevena	GS1 Bosnia & Herzegovina
Kungl, Jens	METRO Group

Name	Organisation
Lam, Ken	GS1 Hong Kong
Lockhead, Sean	GS1 GO
Maatsubatashi, Roberto	GS1 Brasil
Muller, Dan	GS1 Switzereland
Ogandzhanov, Georgy	GS1 Russia
Olsen, James	MultiAd Kwikee
Oney, Ilteris	GS1 GO
Osborne, Andrew	GS1 UK
Paixao, Silverio	GS1 Portugal
Peter, Bijoy	GS1 India
Ramos, Carlos	GS1 Mexico
Ramos Velazquez, Emmanuel	GS1 Mexico
Richardson, Rich	GS1 US
Rivarola, Sebastian	Eway S.A.
Robba, Steven	1WorldSync Holdings, Inc.
Rosell, Pere	GS1 Spain
Rutger, Maciej	GS1 Poland
Sahni, Vipin	GS1 India
Sehorz, Eugen	GS1 Austria
Sheldon, David	Nestle
Soboleva, Olga	GS1 Russia
Strouse, Owen	GS1 Global Office
Swaminathan, Sachidanantham	GS1 India
Tluchor, Tomas	GS1 Czech Republic
Troeger, Ralph	GS1 Germany
van den Bos, Frits	GS1 Netherlands
Vuckovac, Denis	Auto-Id Labs ETH Zurich
Walker, John	Semaku
Wiesmann, Matthias	Google Switzerland GmbH
Yu, Shi	Beijing REN JU ZHI HUI Technology Co. Ltd.
Zhang, Tony	FSE, Inc.
Zuniga, Sergio	GS1 Mexico
Zurn, Darryl	Smiths Medical

3 Log of Changes

Release	Date of Change	Changed By	Summary of Change
1.0	June 2015	Eric Kauz	Initial release
1.0.1	Nov 2015	David Buckley	GTIN+ on the web renamed to GS1 SmartSearch

Disclaimer

GS1®, under its IP Policy, seeks to avoid uncertainty regarding intellectual property claims by requiring the participants in the Work Group that developed this **GS1 SmartSearch Implementation Guideline** to agree to grant to GS1 members a royalty-free licence or a RAND licence to Necessary Claims, as that term is defined in the GS1 IP Policy. Furthermore, attention is drawn to the possibility that an implementation of one or more features of this Specification may be the subject of a patent or other intellectual property right that does not involve a Necessary Claim. Any such patent or other intellectual property right is not subject to the licencing obligations of GS1. Moreover, the agreement to grant licences provided under the GS1 IP Policy does not include IP rights and any claims of third parties who were not participants in the Work Group.

Accordingly, GS1 recommends that any organisation developing an implementation designed to be in conformance with this Specification should determine whether there are any patents that may encompass a specific implementation that the organisation is developing in compliance with the Specification and whether a licence under a patent or other intellectual property right is needed. Such a determination of a need for licencing should be made in view of the details of the specific system designed by the organisation in consultation with their own patent counsel.

Table of Contents

1 Introduction

1.1 Purpose of this Document

This document provides guidance about how machine-readable structured data about a product or product offering can be embedded within an existing web page. “Data about a product” refers to information that describes the product, such as its name, physical dimensions, ingredients, suggested use, and so on. “Structured data” refers to data that is not just free-form text, but rather is organised into individual units of data, often called “data elements” or “attributes,” and that the same data elements are used in a consistent way to describe many different products. Structured data about products is often called “product master data,” the term “master” suggesting that such data is the foundation for many different data processing tasks that may need to understand the attributes of a product. Such tasks include goals primarily of value to businesses (“internal” or “B2B” applications) and also goals of value to consumers (“B2C” applications).

For many years, there have been GS1 Standards that define product master data, principally the standards comprising the Global Data Synchronisation Network (GDSN). These standards include definitions of thousands of product data attributes, the Global Product Classification standard (GPC), and standards for the B2B exchange of product master data over the public Internet. However, these standards have been primarily aimed at meeting the needs of B2B applications within the supply chain, especially the communication of product master data from manufacturers to retailers in order to automate various business processes in the order-to-cash process between those parties.

The focus of this document, in contrast, is structured data for use in B2C applications. There are key differences between B2C applications and the sort of B2B applications towards which GDSN is aimed:

- The set of data attributes required by B2C applications differs from B2B data attributes, although there are significant areas of commonality.
- The approach to delivering structured data for B2C applications is based the open interaction model of the World Wide Web, not the closed point-to-point approach in GDSN.
- B2C applications require integrating data from multiple sources.

There are many facets to the overall problem of providing structured product data for B2C applications. This document focuses specifically on how structured product data may be embedded into public-facing Web pages that present products and information about products to consumers. Other GS1 Standards address other facets of the B2C problem; for example, the GS1 Source (Trusted Source of Data) standard addresses the distribution of B2C data to Internet application providers via a network of known “data aggregators” that act as hubs for the distribution of data about products from many brand owners.

The technical approach described in this document, in contrast, is based on:

- Resource Description Format (RDF) as the language for expressing structured data
- Schema.org and GS1 vocabularies to populate the structured data
- JavaScript Object Notation for Linked Data (JSON-LD) as the machine-readable syntax for encoding the structured data into a format that can be easily embedded into a web page. Compared to alternative syntaxes for RDF (including Microdata, RDFa, and Microformats), JSON-LD has the advantage of allowing the structured data to be inserted into an existing web page as a single block of text.

These technologies allow structured product data to be inserted directly into public-facing Web pages, where it is available to any application that consumes those pages. This allows web page publishers to distribute product data directly to consumers of the web page.

The data aggregator-based approach of GS1 Source and the web page approach described in this document are complementary. GS1 Source is designed to address the needs of applications that need reliable, authoritative access to product data about a large range of products. Mobile applications that scan a barcode or search for a product are good examples of these. The web page approach described in this document is designed to address the needs of applications that need deeper insight into the content of a particular web page that the application happens to be

consuming. Web search engines such as Google, Bing, and others are very important examples of such applications; social media sites, shopping engines, and other emerging applications are others.

While the ultimate aim of this document is to provide technical guidance, the document also includes background and business motivation, too.

1.2 Who Will Use this Document?

Brand Owners, Manufacturers, Retailers, Advertising Agencies and Search Engine Optimisation (SEO) Strategists can benefit from providing structured data about products and product offerings in order to improve the visibility / discoverability of those products or offerings on the web. Near-term benefits include enhanced search results such as "rich snippets" presented by search engines such as Google and Bing, and recognition of products by social media platforms that are already prepared to process structured data. However, the provision of structured data about products and offerings from trusted authoritative sources (such as the brand owner or retailer) also serve to build a data platform of Linked Data that can support a large number of novel smartphone applications (apps) that are related to products and provide consumers with the information they need to:

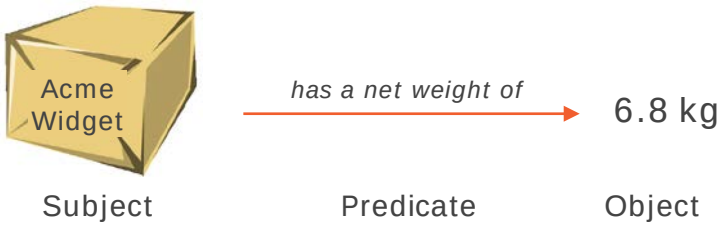
- Choose products that best suit their needs
- Access services, such as instruction manuals, recipes, troubleshooting guides, warranty registration, information about related products, accessories and consumables.

Section 2 provides further details about the business motivation for implementation.

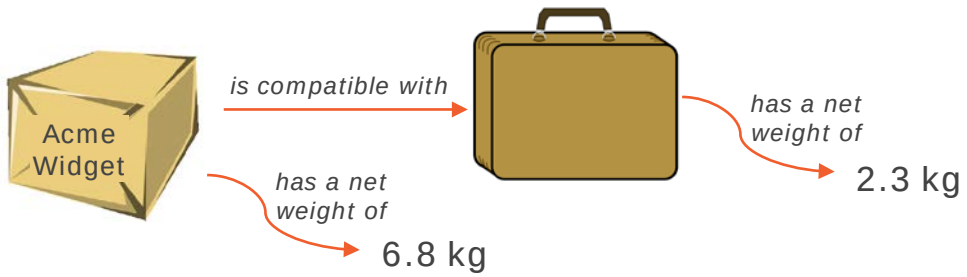
1.3 Brief Introduction to Linked Data about products and offerings

Product master data can be thought of as information that connects a product with its Global Trade Item Number (GTIN), its brand name, product name, description, net weight, technical data, ingredients list or material composition, nutritional information, usage instructions, etc. Some properties are generally applicable to all products. Others (such as nutritional information) are only applicable to specific sub-classes of products, such as Food / Beverage / Tobacco products.

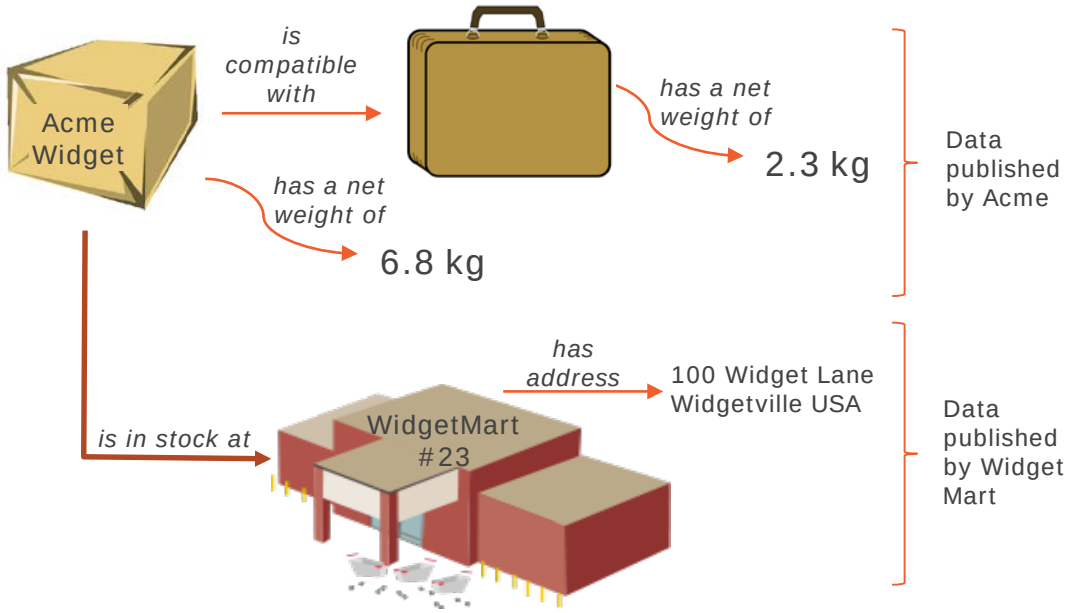
Linked Data technology provides a mechanism for exchanging and linking such structured data about products on the web. In Linked Data, each relationship between a thing and an attribute that describes the thing is expressed as *triple* of Subject – Predicate – Object; for example, "The Acme Widget product (subject) has a net weight (predicate) of 6.8 kilograms (object)." The formal computer language for information expressed in this form is called Resource Description Format (RDF) and each Subject – Predicate – Object relationship is simply called an "RDF Triple."



The Object of one RDF triple might be the Subject of another. For example, in the triple "The Acme Widget product (subject) is compatible with (predicate) the Acme Widget Blue Carrying Case (object)" the Object of this triple (the carrying case) could well be the Subject of many other triples that provide information about the carrying case.



The power of Linked Data arises not only from the ability to link RDF Triples together, but also because linked RDF Triples may come from many sources. For example, the manufacturer of Acme Widgets might publish the triples illustrated above, and independently a retailer might publish another triple "The Acme Widget product (subject) is in stock (predicate) at WidgetMart Store #23 (object)". The information about the product published by the manufacturer is now linked with the information about product availability published by the retailer.



When we want to provide this structured data in a machine-interpretable way, we need to use HTTP URIs to identify the Subject, Predicate and any Object that is not a simple value such as a text string, date/time or number. We use web vocabularies or ontologies to choose concepts, categories or 'classes' and associated properties, attributes or 'predicates' defined in such web vocabularies or ontologies, each associated with an HTTP URI for a specific predicate or class.

For the Subject of such 'facts' about a product or product offering, the brand owner / manufacturer or retailer needs to create HTTP URIs based on one of their own registered domain names. The HTTP URIs for Predicates are usually taken from existing web vocabularies or ontologies, such as Schema.org, GoodRelations or the new GS1 web vocabulary, in order to use specific properties or attributes (e.g. price, weight, description) that are defined in such vocabularies or ontologies. In Section 3.1 and Section 3.2, we explain how brand owners / manufacturers and retailers can construct such HTTP URIs for identifying the product or product offering as the Subject of such 'facts'.

We can use markup formats such as JSON-LD, RDFa or Microdata to embed the structured data in web pages. We then need to serve the Linked Data, either directly or by embedding it within an existing web page.

1.4 HTTP URIs in Linked Data

The following notes should be read as background to both of these sections.

HTTP URIs can be used to identify both information resources (such as web pages) and non-information resources (such as products, people, places and organisations in the real world). Given an HTTP URI, it is easy to access information simply by making an HTTP request (web request). Beyond what is specified in the URI standard, the structure and syntax of an HTTP URI is opaque, which means that by inspection of the URI alone, it is generally not possible to know whether an HTTP URI serves to identify an information resource (such as a web page) or a non-information resource (such as a real-world product).

Linked Data technology uses HTTP URIs to identify information resources (such as web pages or collections of facts about a product) and allow those to be retrieved via a web request. Linked Data technology also uses HTTP URIs in the expression of facts about real-world things; facts are represented as RDF triples consisting of a Subject, Predicate, Object. The Subject is the topic about which the facts are expressed. The Predicate represents a specific Property or Attribute of the Subject or its Relationship to another thing. The Object represents the value of the specified Predicate, which may be a simple literal value, such as a Text String, DateTime timestamp or Number – or a complex data object that has further attributes or properties of its own. A very simple example of a complex data object is a MeasurementType, which consists of a numeric value and a unit of measure.

When we want to write factual information about a product or product offering, it is a good idea to choose an HTTP URI that we can use as the Subject of a number of RDF triples. Such an HTTP URI represents a non-information resource; it represents the product itself. Using an HTTP URI means that via a web request we can retrieve (dereference) a collection of RDF triples for which that HTTP URI is the Subject (or possibly the Object) and others can also re-use the same HTTP URI in facts that they write about the same Subject. For example, a brand owner may construct an HTTP URI for each of its product GTIN values and retailers that sell those products can quote those HTTP URIs verbatim in facts that they write about their offerings for the product. In this way, the retailer can write Linked Data that expresses that a specific product offering includes a specific product from a particular brand owner or manufacturer. In this way, it is not strictly necessary for the retailer to replicate or embed all of the facts about the product that were asserted by the Brand Owner, since these are usually retrievable via a web request for the HTTP URI constructed by the brand owner.

2 Business Motivation for Deploying Linked Data About Products

2.1 Why should I introduce structured data on my web site?

There are a number of different motivations for including structured data within your web site. The most common are:

- Provide easier access to information about your products and product offerings to consumers through improved search results and “rich snippets”
- Expose information about your products and product offerings in a way that enables it to be consumed by applications and mobile apps
- Expose information in a way that will enable it to be linked to other sources of information about the same product e.g. to link together information from brands, retailers, consumers and other parties
- Provide a way to link between physical products and their identity on the web (by means of QR code and mobile scanning)

2.2 Is this new technology?

No. Many brands and retailers have already introduced some structured data within their websites in order to enhance the search results for their products and product offerings. The GS1 Web Vocabulary uses this existing technology to extend the range of product attributes that can be

included within your web pages by reusing the standard definitions that already exist within the GS1 data dictionary and global product classification.

2.3 Why is GS1 involved and why have they developed the GS1 Web Vocabulary?

Over the years the GS1 community have developed standard ways to represent information about products in order to support members in their ability to introduce new products and manage their end-to-end supply chain. The GS1 SmartSearch initiative was developed to enable these existing standards to be used to expose consumer facing product information directly within web pages. It is anticipated that the GS1 Web Vocabulary for product information will enable brand owners and retailers to describe product characteristics in richer detail than they can currently do using existing ontologies such as *schema.org* or *GoodRelations*. By including GS1 keys and classifications (GTIN and GPC) it is anticipated that users will facilitate the linking of data for products and offers between retailer, manufacturer, search provider and other sites.

2.4 I already use Schema.org to describe my products. Why should I use the GS1 Web Vocabulary?

The GS1 Web Vocabulary is intended to complement existing ontologies. It is anticipated that web sites will use both (you can use both within the same page). Reasons to use the GS1 Web Vocabulary are:

- It is more detailed so enables you to assert more facts about your products and offerings. For example: It is fully aligned to the latest EU regulation regarding the online disclosure of product information to consumers (EU 1169/2011).
- Attributes within the GS1 Web Vocabulary are derived directly from existing GS1 standard terms which will assist companies who are already using GS1 standards

Note that where an attribute in the GS1 Web Vocabulary already exists within Schema.org this is expressed using a *'sameAs' statement* to provide a mapping between the terms.

2.5 What are the benefits for brand owners and manufacturers?

Brand owners can benefit from Linked Data because this will ensure that their products are highly visible on the web (including detailed product specifications, ingredients, nutritional information, health, environmental and ethical accreditations, as well as links to technical datasheets, instruction manuals and online help). Using Linked Data technology, small manufacturers of specialised niche products can achieve the same web visibility of their products as larger manufacturers of mass-market products. By using structured data manufacturers and brands can make unambiguous and authoritative information about their products directly accessible to consumers, thus reducing the risk that consumers will rely on potentially poor quality or inaccurate/outdated information from alternative sources.

Additionally, by providing rich structured data about products and product offerings, brand owners can improve insight into the search criteria that consumers are using to find and select the products that best match their needs; the open availability of rich structured data enables a range of new consumer-facing applications for product search and comparison – these may be in the form of smartphone apps or in-store consumer apps. This in turn can provide brand owners with information about consumer behaviour and preferences that enables them to improve their products to focus on improving the criteria of interest to consumers (e.g. particular product specifications or environmental / ethical accreditations).

2.6 What are the benefits for retailers?

Retailers can benefit from Linked Data technology by ensuring that their product offerings are highly visible on the web (including details of promotions and special offers, availability, customer reviews and ratings, cross-selling opportunities for related products and accessories). Linked Data can be used to provide an enhanced customer experience on the web and via retailer smartphone apps or in-store consumer apps. For example, some retailers have already developed digital shopping list

apps that enable consumers to make intelligent choices about products, based on their individual dietary requirements, or preferences on health, environmental or ethical issues.

Additionally, by providing rich structured data about products and product offerings, retailers may gain insight into the search criteria that consumers are using to find products and select the products that best match their needs; the open availability of rich structured data enables a range of new consumer-facing applications for product search and comparison – these may be in the form of smartphone apps or even be installed within in-store scan-as-you-shop handheld terminals for customer use. This in turn can provide retailers with information about consumer behaviour and preferences that enables them to give higher priority to stocking products that match the particular criteria of interest to consumers (e.g. particular product specifications or environmental / ethical accreditations)

2.7 What are the benefits for consumers?

The introduction of structured data will benefit consumers by:

- Giving them more accurate and helpful search results for products that meet their needs e.g. a search result for a product that meets my dietary needs, near me now, provided with a suitable rich snippet of information about the product and where it can be bought
- Making it easier to carry out side-by-side comparison of products because the meaning of data is less ambiguous than plain text within a page
- Helping consumers have confidence in the trustworthiness of product data (that brands choose to publish)
- Giving consumers new ways to access product information, by search or QR scanning, using applications and apps that analyse and present information in a standard way regardless of brand or retailer e.g. nutrition and ingredients.

2.8 Who publishes the structured data?

It is envisaged that brand owners will choose to publish facts about their products (GS1 trade items) and that retailers will publish facts about their product offerings (and often reference or include facts asserted by the brand owner)

2.9 Who can consume the structured data?

Including structured data within a web page makes it accessible to any machine with access to that page. Typical consumers of structured data are search engines and data aggregators —but could be anyone or any other piece of software that wishes to make use of the data.

2.10 How can structured data for a product be accessed using a QR code?

By including an HTTP URI such as `http://id.examplebrand.com/gtin/05011476100885` within a QR code on a product, a scanning app will be directed to the product's identity on the web. Depending upon the type of request made by the app it will served either with structured data about the product or will be re-directed to any normal web page relating to the product. It is therefore possible to serve the needs of both existing QR scanning applications and new applications that may want to process structured data relating to the product.

2.11 How will information about retail product offerings be made available to consumers via search and apps?

Linked data will enable search engines and other web information providers to blend together information about products about retailer offers in a way that best fits the requirements and context of a given request. For example: A picture and description of a product (from the brand) merged with nearby retailer offerings of that product its availability and delivery/collection options.

2.12 How do I get started?

If you have no prior experience with structured data then the following practical suggestions may help guide your first steps.

Establish the opportunity within your company

- Understand how your company currently makes information about its products or offers available on the web (including how accurately or prominently this information can be found within web search results).
- Find out whether your company is already using structured data within its web pages, either from your web site team or using freely available tools such as:
  - http://www.google.com/webmasters/tools/richsnippets
  - http://linter.structured-data.org/
- Examine the information displayed on your website and how it maps to the schema.org and GS1 vocabularies.
- Experiment with adding structured data to your pages, using the schema.org and GS1 vocabularies, to improve your understanding and measure the impact that this has upon search and the visibility of your product or offer. Initially this could be for just a small number of attributes such as GTIN, product or offer description and image. (You should strongly consider including GTIN if you want to make it easy for search and other applications to associate structured data on your web pages with other relevant data.)

Build knowledge and capability within your company

- Promote and build awareness of the potential benefits of structured data with your business commercial and marketing teams using the information in this implementation guide and any of your own experience.
- Share the technical sections of this guide with your web developers and understand whether they are already familiar with or using any of the related standards such as RDFa, schema.org, JSON-LD. Encourage them to experiment and become familiar with the standards and technology.

Provide feedback and get involved in the development of GS1 standards for the web

- Review and provide feedback on the GS1 web vocabulary and this guide based upon your own experience.
- Get involved with the GS1 working groups that maintain this guideline and the related GS1 Standards. There are also industry associations that do related work.

2.13 What training and education do GS1 Member Organisations need to develop? Who gives guidance?

We recommend that GS1 Member Organisations co-ordinate very closely with GS1 SmartSearch activities sponsored by the GS1 Global Office, to avoid duplication of effort or conflicting information.

The GS1 SmartSearch MSWG is already committed to developing guidance material about how to publish Linked Data for products - including how to use the GS1 Web Vocabulary and HTTP URIs for products and product offers, incorporating the GTIN and other qualifiers (e.g. Serial Number, Lot/Batch) where further granularity is required.

2.14 What are the challenges going to be in terms of getting retailers and brands on board?

Today – many brands and retailers have already introduced structured data within their websites in order to improve search results. But many companies are only just beginning to understand how the web is evolving and the benefits of embracing semantic / linked data technologies. For many, the initial incentive may be the opportunity to achieve enhanced search results, such as “rich snippets,”

whereas a few are considering the longer term benefits and the new kinds of opportunities and product-related services that could be enabled in the next generation of smartphone apps if rich structured data about products is readily available on the web. As with any new technology, there can be a 'first mover advantage' but some companies are still unclear how to proceed with new unfamiliar technology – or are concerned about whether they somehow relinquish control if they publish such data using Linked Data technology. The fact is that initially, much of the structured machine-processable data will not be data that is commercially sensitive (such as traceability data) but information that is already effectively in the public domain because it appears on the packaging or products or in human-readable format in public web pages. For its part, GS1 is trying to educate its users about Linked Data technology, the potential benefits – and provide not only a GS1 web vocabulary that is aligned with the precise definitions of terms developed by the GS1 community through a consensus process spanning several decades – but also to provide tools such as JSON-LD templates that should make it much easier for companies to adopt Linked Data technology. GS1 is also encouraging and supporting a number of pilot activities in this area.

2.15 Why is it attractive to marketing / SEO agencies?

For marketing agencies, Linked Data technology helps to fine-tune web search results and helps consumers to find exactly the products and services that are of interest to them, because the detailed product characteristics become more readily available to search engines and smartphone apps. Advertising agencies that become deeply familiar with semantic / Linked Data technologies will be in the strongest position to provide the greatest value to clients as the web evolves.

2.16 How will search engine listings be impacted by structured data?

Using the schema.org, GoodRelations or GS1 vocabularies, it is possible to associate rich structured data about products with the globally unique identity of the product (its Global Trade Item Number or GTIN – typically the number on the barcode) by using properties such as `http://schema.org/gtin13`.

The use of the GTIN provides a consistent cross-reference between information provided by the brand owner and multiple retailers and resellers of the product. It enables search engines to quickly determine which data about a product is consistent – and which information is likely to be misleading.

Furthermore, GS1 is working on a mapping of its product classification systems (such as GPC Global Product Classification) to a Linked Data representation, which means that consumers will be able to more reliably find products that match particular categories and criteria / attributes even when the consumer searches by keyword and does not have a specific brand in mind.

Search engines including Google and Bing use structured data in order to produce "rich snippets." These are the enhanced search listings that typically appear on the right-hand side of the search results page, often including photos, maps, opening hours, price information (or in the case of music bands, lists of their albums and upcoming concerts).

Rich snippets may draw upon information from multiple sources - there is not always an exact 1-1 relationship with the structured data added to the website - and they might only display a subset of the information or complement it with information from elsewhere (e.g. blending product master data from the brand owner with information about the offering (price, promotions etc.) from the retailer). So in the case of music bands, the rich snippet may be a blend of information from the band's own web page and sites such as MusicBrainz and Wikipedia.

However, for products, adding structured data to an existing web page is probably the best method, although it is also possible to provide new 'pages' (scripts or servlets) that serve only the data about products, e.g. as JSON-LD or *RDF Turtle* even if there is no corresponding web page.

2.17 How will the search engine surface products searched under 'red shoes' for example?

Within GS1, we have the Global Product Classification (GPC) system, although currently very few brand owners and retailers include the GPC brick values and attribute-value pairs within the markup for their web pages. We have done an initial crude translation of GPC into RDF format, but need to do some further work to make it more web-developer friendly, avoiding the need for them to

understand the current 8-digit GPC codes, some of which appear to have been assigned on a first-come-first-served basis.

In parallel, GS1 US is working with some major companies on the design of a Structured Commerce Classification code.

We expect that a developer-friendly GS1-endorsed product classification system suitable for use with Linked Data will emerge from these efforts and that when brand owners and/or retailers make use of this, searches by product category and attribute will become easier using Linked Data. The GS1 Web Vocabulary will also include some support for product categories and attributes, with particular support for quantitative attribute-value pairs (as currently expressed in the GDSN data model, whereas GPC is primarily concerned with qualitative attribute-value pairs).

2.18 Will the semantic web become integral to search engine optimisation (SEO)?

Linked Data technologies are already becoming integral to SEO. Even outside of retail and consumer products, organisations and individuals are improving their own SEO by using semantic technology. For example, many musicians and bands are already benefitting from Google Rich Snippets because they have put structured data about their discography into MusicBrainz, a biography into Wikipedia, their upcoming concert listings into BandsInTown or SongKick and they have cross-referenced between these sites and also linked to their websites and channels on various social media networks (e.g. Facebook, YouTube, Twitter), which means that search engines identify the connections across these constellations of linked data and begin to recognise them as 'Named Entities' with interesting facts and figures – and the bands or musicians benefit from enhanced search results such as rich snippets.

2.19 How will search engines know who the trusted source of data is?

The structured data includes property tags such as `http://schema.org/brand` so if for example a brand owner page at nestle.com includes that property tag that points to a data object of type `http://schema.org/Brand` and itself having a `http://schema.org/name` property of "Nestlé", then a search engine should be fairly confident that the data is coming from that brand owner.

If a retailer such as Tesco has a web page about a Nestlé product (e.g. KitKat), they can also include structured data markup to say that the product in their offer is from the brand "Nestlé" - which might result in a search engine merging some structured data provided by the brand owner with other structured data from the retailer (e.g. about price and availability or retailer promotions)

At a technical level, it is also possible to use Data Provenance standards from W3C (http://www.w3.org/standards/techs/provenance#w3c_all) to express the source of the data and how it has been transformed. Usage of JSON-LD or quads rather than triples make it very easy to attach metadata assertions about authorship to the structure data.

2.20 How and will sponsored search results, such as Google AdWords, be impacted?

It's hard to predict. *Google AdWords* and similar offerings from other search engines are commercial offerings from those search engines to promote search results to a higher position when the search query contains those words. At this time, no search engine has indicated if or how it intends to use Linked Data to affect sponsored search results.

2.21 How do we engage the web development world?

The GS1 Web Vocabulary and guidance material mentioned above will be published, to provide sufficient information to web developers about how they can make use of the rich structured data about products.

2.22 What are the implications for merchants' product data feeds to search engine marketplaces?

GS1 is in discussions with Google about their Google Shopping merchant feeds and how we can map from the GS1 Web Vocabulary to the characteristics they expect in the Google Shopping merchant feeds. It is conceivable that we or they might develop some translation tools so that data marked up with the GS1 Web Vocabulary can easily be transformed into the expected markup for the Google Shopping merchant feeds, even if Google and GS1 currently use slightly different terminology for some attributes that are semantically equivalent.

2.23 Walk me through how search could be enhanced in future, based upon linked data

The more widespread the use of linked data becomes the more valuable it will become as an aid to consumers who are searching for products and product offers. An illustration of this is show below:

- A consumer enters a general search term into an app or search engine
- The search term is matched against the global product classification (GPC) or similar to reveal the product attributes and values relevant to the class of products being searched for
- The consumer is offered the option to refine their search by setting values of the attributes provided and specifying additional criteria, e.g. their budget, location or how urgently they need the product
- Products (GTINs) that match the GPC and attribute-values provided are considered. (The GTIN identifier links to additional information such as product specifications, weight, ingredients, nutritional information, image or description).
- Retailers offering these GTINs can be identified (which provides a link to the retailers price, current availability etc).
- Linked data about the location of retailers can be used to display products meeting the original search criteria on a map.

3 Implementation Procedures

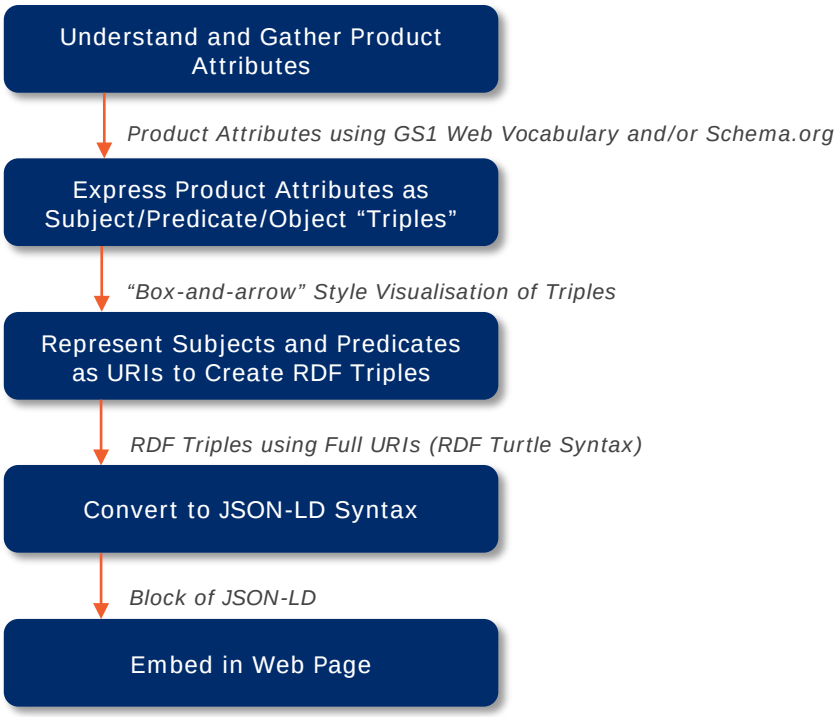
This section explains procedures for implementing structured linked open data about products or product offerings, using a single block of JavaScript Object Notation for Linked Data (JSON-LD) embedded within a web page, either within the header (<head> section) or at the end of the <body> section.

Appendix A provides the essential technical background about Linked Data / Semantic Web and specifically about HTTP URIs, JSON-LD and the GS1 Web Vocabulary.

Brand Owners / Manufacturers should consider the implementation procedures detailed in Sections 3.1, 3.3, 3.5, and 3.7 through 3.11.

Retailers should consider the implementation procedures detailed in all sections 3.1 through 3.11.

The process of creating linked data is somewhat complex, so it is best to conceive of the process in stages:



3.1 Procedure for brand owners or manufacturers to construct an HTTP URI to identify a product in facts asserted using Linked Data

A brand owner or manufacturer wishes to construct an HTTP URI within one of their own registered domain names for the purpose of identifying a non-information resource (e.g. a physical product or digital product) so that they can assert facts about that object as Linked Data.

3.1.1 Pre-Requisite

We advise reading Technical Appendix A in order to familiarise yourself with the essentials of Linked Data technology.

3.1.2 When Would I Use This?

A brand owner or manufacturer should create one HTTP URI per product GTIN for each product GTIN they issue. If additionally they also wish to be able to write facts or serve data at finer granularity than the GTIN (e.g. GTIN + Lot/Batch or GTIN+Serial Number), then they should additionally create HTTP URIs based on those combinations of GTIN and other qualifiers. Typically this procedure is only performed once, to define a pattern for constructing HTTP URIs for a given GTIN (or GTIN + qualifiers).

3.1.3 How To?

A brand owner or manufacturer uses an existing Internet domain name registered to it – or registers a new domain name specifically for this purpose. It is a good idea to use relatively short domain names if the corresponding URLs will be used with QR codes, since short URLs require fewer pixels for encoding in a QR code, resulting in a code that is more 'chunky' and easier to read at a distance or when the optical resolving power of some smartphone camera optics is relatively low.

From this domain name, the brand owner constructs an HTTP URI pattern to be used for each of its products.

For example, if a brand owner currently leases the domain name `brandexample.com`, they might construct HTTP URIs such as:

http://id.brandexample.com/gtin/00614141123452

or

http://brandexample.com/id/gtin/00614141123452

Of course from a Linked Data perspective, there is no requirement that the GTIN value should appear within the HTTP URI, whether that URI is used in RDF triples or encoded within a QR code, Near-Field Communication (NFC) tag, or other data carrier. Including the GTIN within the URI is merely a convenience for the brand owner or manufacturer, to make it easier to avoid duplication and to remember which URI is intended to refer to which product.

A separate document will provide guidance about encoding of HTTP URIs in QR codes (either regular QR codes or GS1 QR codes). For use with GS1 QR codes in which the GTIN and other qualifiers (such as Lot/Batch number or Serial Number) are separately encoded using GS1 Application Identifiers, such guidance is likely to propose a mechanism for constructing a virtual canonical HTTP URI from a short HTTP URI prefix together with the GTIN and other qualifiers. However, that virtual canonical HTTP URI might never be written in any facts and might only be configured (via the URI rewriting rules of a webserver) to redirect to the HTTP URI that a brand owner prefers to use.

It should also be noted that the appearance of a GTIN (or other qualifiers) within an HTTP URI should not be interpreted as a reliable assertion that the URI represents a product or product offering with a specific GTIN or links to information about a product or product offering with that GTIN. The only reliable way to draw that conclusion is if there is a specific RDF triple that asserts that the Subject has a specific GTIN value. One way to do this is by using the schema.org Predicates: http://schema.org/gtin13 or http://schema.org/gtin14.

3.2 Procedure for retailers to construct an HTTP URI to identify a product offering in facts asserted using Linked Data

A retailer wishes to construct an HTTP URI within one of their own registered domain names for the purpose of identifying a non-information resource (e.g. an offering for a particular product) so that they can assert facts (such as price, availability and promotional offers) about that offering as Linked Data.

3.2.1 Pre-Requisite

We advise reading Technical Appendix A in order to familiarise yourself with the essentials of Linked Data technology.

3.2.2 When Would I Use This?

A brand owner or manufacturer should create one HTTP URI per product GTIN for each product GTIN they offer for sale. If additionally they also wish to be able to write facts or serve data at finer granularity than the GTIN (e.g. GTIN + Lot/Batch or GTIN+Serial Number), then they should additionally create HTTP URIs based on those combinations of GTIN and other qualifiers. Typically this procedure is only performed once, to define a pattern for constructing HTTP URIs for a given GTIN (or GTIN + qualifiers).

3.2.3 How To?

A retailer or reseller uses an existing Internet domain name registered to it – or registers a new domain name specifically for this purpose. It is a good idea to use relatively short domain names if the corresponding URLs will be used with QR codes on retailer-specific packaging, since short URLs require fewer pixels for encoding in a QR code, resulting in a code that is more 'chunky' and easier to read at a distance or when the optical resolving power of some smartphone camera optics is relatively low.

From this domain name, the retailer constructs an HTTP URI pattern to be used for each of the products it offers for sale.

For example, if a retailer currently leases the domain name retailerexample.com, they might construct HTTP URIs such as:

628 `http://id.retailerexample.com/gtin/00614141123452`

629 or

630 `http://retailerexample.com/id/gtin/00614141123452`

631 Of course from a Linked Data perspective, there is no requirement that the GTIN value should
632 appear within the HTTP URI, whether that URI is used in RDF triples or encoded within a QR code,
633 NFC tag, or other data carrier. Including the GTIN within the URI is merely a convenience for the
634 retailer, to make it easier to avoid duplication and to remember which URI is intended to refer to
635 which product.

636 A separate document will provide guidance about encoding of HTTP URIs in QR codes (either regular
637 QR codes or GS1 QR codes). For use with GS1 QR codes in which the GTIN and other qualifiers
638 (such as Lot/Batch number or Serial Number) are separately encoded using GS1 Application
639 Identifiers, such guidance is likely to propose a mechanism for constructing a virtual canonical HTTP
640 URI from a short HTTP URI prefix together with the GTIN and other qualifiers. However, that virtual
641 canonical HTTP URI might never be written in any facts and might only be configured (via the URI
642 rewriting rules of a webserver) to redirect to the HTTP URI that a retailer prefers to use.

643 It should also be noted that the appearance of a GTIN (or other qualifiers) within an HTTP URI
644 should not be interpreted as a reliable assertion that the URI represents a product or product
645 offering with a specific GTIN or links to information about a product or product offering with that
646 GTIN. The only reliable way to draw that conclusion is if there is a specific RDF triple that asserts
647 that the Subject has a specific GTIN value. One way to do this is by using the schema.org
648 Predicates: `http://schema.org/gtin13` or `http://schema.org/gtin14`.

649 3.3 Procedure for brand owners or manufacturers to construct a simple block 650 of JSON-LD to represent basic facts about any product, using the 651 schema.org vocabulary

652 A brand owner or manufacturer wishes to include create linked data about their product so that they
653 can assert basic facts about that product such as the product's name, description, and image, and
654 enable others (e.g., retailers) to link to this information.

655 3.3.1 Pre-Requisite

656 Section 3.1.

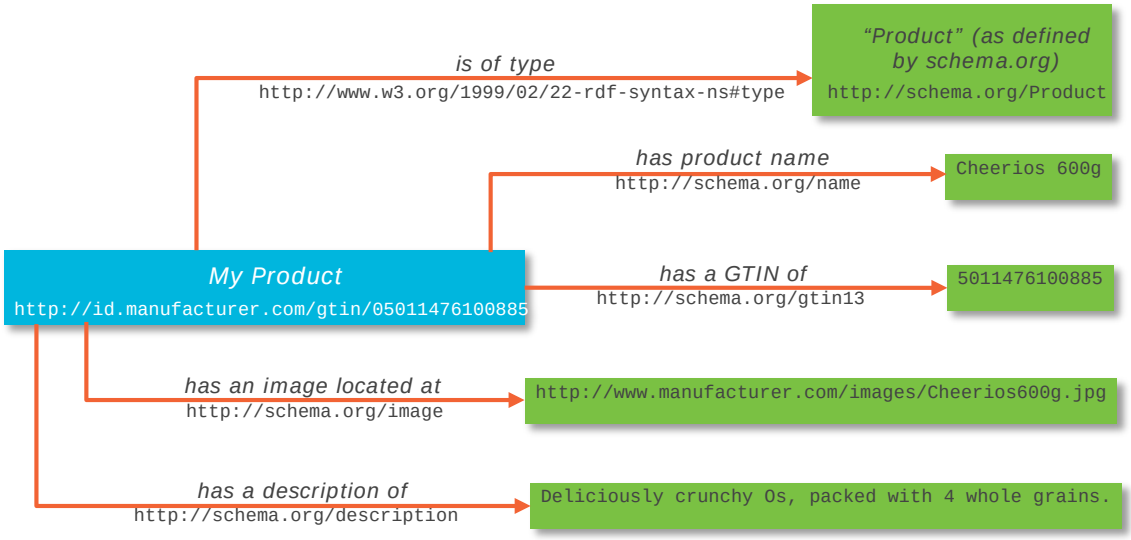
657 3.3.2 When Would I Use This?

658 Use this procedure when creating linked data for your product using the schema.org vocabulary.

659 3.3.3 How To?

660 The JSON-LD snippet below is a very minimal example showing how a brand owner or manufacturer
661 could use the schema.org vocabulary to write some simple facts about a product and mark them up
662 as a single block of JSON-LD.

663 Start with a visualisation of the facts we want to write:



In this figure, concepts are written in italics, and the URI representation of those concepts as used in RDF written below that.

This corresponds to the following set of RDF triples, written below in *RDF Turtle* notation:

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix schema: <http://schema.org/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://id.manufacturer.com/gtin/05011476100885> rdf:type schema:Product .
<http://id.manufacturer.com/gtin/05011476100885> schema:gtin13 "5011476100885" .
<http://id.manufacturer.com/gtin/05011476100885> schema:name "Cheerios 600g"@en .
<http://id.manufacturer.com/gtin/05011476100885> schema:image
<http://www.manufacturer.com/images/Cheerios600g.jpg> .
<http://id.manufacturer.com/gtin/05011476100885> schema:description "Deliciously
crunchy Os, packed with 4 whole grains."@en .
```

The first four lines of the RDF Turtle notation define namespace prefixes that are used in the remainder, and the remaining lines of RDF Turtle contain the triples. Each triple is simply written as Subject Predicate Object, separated by spaces and terminated by a period.

Writing RDF Turtle is a good intermediate step because it makes the RDF triples very clear. But to embed in a web page, you next have to translate this into a web-compatible format. This guideline recommends JSON-LD as such a format, as it allows the structured data to be inserted into a web page in a single block rather than being interspersed with the human-facing content. Here is how the above triples look when written in JSON-LD:

```
{
  "@context": {
     "schema": "http://schema.org/",
     "xsd": "http://www.w3.org/2001/XMLSchema#",

     "Product":"schema:Product",
     "gtin13": {"@id":"schema:gtin13", "@type": "xsd:string" },
     "name":{"@id":"schema:name", "@language": "en"},
     "description":{"@id":"schema:description", "@language": "en"},
     "image":{"@id":"schema:image", "@type":"@id"}
  }

 ,
  "@id": "http://id.manufacturer.com/gtin/05011476100885",
  "@type": [ "Product"],
  "gtin13":"5011476100885",
```

```
  "name": "Cheerios 600g",
  "description": "Deliciously crunchy Os, packed with 4 whole grains.",
  "image": http://www.manufacturer.com/images/Cheerios600g.jpg
}
```

The full JSON-LD block has two parts: the context (shaded orange) and the body (shaded blue). The body contains the structured data we wish to publish, and the context sets up abbreviations used in the body so that the body is easier to read and process. (In this example, it may not look like the context is saving us much. But a block of JSON-LD embedded in a real web page could contain much more data and the benefit of the context would more obvious.)

Let's first consider the body part of the JSON-LD block (shaded blue). The general structure is a set of *property : value* pairs. Mostly, in each pair the *property* is an RDF predicate (interpreted with the help of the context) and the *value* is an RDF object. But there are some special pairs, too.

Let's consider it line by line:

```
  "@id": "http://id.manufacturer.com/gtin/05011476100885",
```

The `@id` property says that we are writing triples about a particular URI; *i.e.* that this URI is the subject in all the triples that follow. In this example the URI is the HTTP URI for the product or trade item, constructed by the brand owner or manufacturer, using one of their own registered domain names. See Section 3.1 for some example patterns of how to construct such an HTTP URI.

```
  "@type": [ "Product"],
```

The `@type` property says that the identified thing has a particular type, in this case "Product". Note however, that the context block defines an expansion of the term "Product" to `schema:Product` (and in turn to `http://schema.org/Product`).

Taken together, these first two lines are equivalent to the RDF Turtle triple:

```
http://id.manufacturer.com/gtin/05011476100885 rdf:type http://schema.org/Product .
```

The remaining lines are the product data.

```
  "gtin13":"5011476100885",
```

This asserts that the identified thing has a specific GTIN-13, in this case `5011476100885`. Again, the context block expands the term `gtin13` to `schema:gtin13` (and in turn to `http://schema.org/gtin13`) .

```
  "name": "Cheerios 600g",
```

This asserts that the identified thing has the name "Cheerios 600g", through the `name` property term expanded by the context block to `http://schema.org/name`.

```
  "description": "Deliciously crunchy Os, packed with 4 whole grains.",
```

This asserts that the identified thing has a particular description as shown, through the `description` property term expanded by the context block to `http://schema.org/description`.

```
  "image": http://www.manufacturer.com/images/Cheerios600g.jpg
```

This asserts that the identified thing has an associated image, whose URI is indicated through the `image` property term expanded by the context block to `http://schema.org/image`).

Now let's go back to the `@context` part of the JSON-LD block (shaded orange). It provides abbreviations so that the JSON-LD body (shaded blue) is written using simple name strings but those local name strings are mapped to HTTP URIs of properties or predicates defined in specified web vocabularies or ontologies.

Let's consider the context block line by line:

```
    "schema": "http://schema.org/",
    "xsd": "http://www.w3.org/2001/XMLSchema#",
```

These two lines define namespace prefixes that are used in the remaining lines of the context block.

```
    "Product":"schema:Product",
```

This defines `Product` as an abbreviation for `Product` as defined in the schema.org namespace, so that when `Product` appears in the JSON-LD body it is understood to mean `http://schema.org/Product`.

```
    "gtin13": {"@id":"schema:gtin13", "@type": "xsd:string" },
```

This does two things. First, it defines `gtin13` as an abbreviation for `gtin13` as defined in the schema.org namespace, so that when `gtin13` appears in the JSON-LD body it is understood to mean `http://schema.org/gtin13`. Second, it defines the data type of values of the `gtin13` property to be strings.

```
    "name":{"@id":"schema:name", "@language": "en"},
```

This does two things. First, it defines `name` as an abbreviation for `name` as defined in the schema.org namespace, so that when `name` appears in the JSON-LD body it is understood to mean `http://schema.org/name`. Second, it says that values of the `name` property are written in English.

```
    "description":{"@id":"schema:description", "@language": "en"},
```

This is similar to the declaration for `name`, but applies to the `description` attribute.

```
    "image":{"@id":"schema:image", "@type":"@id"}
```

This does two things. First, it defines `image` as an abbreviation for `image` as defined in the schema.org namespace, so that when `image` appears in the JSON-LD body it is understood to mean `http://schema.org/image`. Second, it defines the data type of values of the `image` property to be identifiers (URIs) which themselves could be subject of other RDF triples.

An important point to note is that we have a free choice of the local property names we use – so for example, we could have written the following JSON-LD and it would still result in the *same* set of RDF triples, even though we have changed all of the local property names compared with the previous example. This is important because if means that if a company is internally using JSON data within their website and using their own local property names, the `@context` block provides a very flexible way to map the local terms to terms defined globally via URIs in web vocabularies and ontologies.

```
{
  "@context": {
"schema": "http://schema.org/",
"xsd": "http://www.w3.org/2001/XMLSchema#",

    "TradeItem":"schema:Product",
    "ean13": {"@id":"schema:gtin13", "@type": "xsd:string" },
    "productName":{"@id":"schema:name","@language": "en"},
    "tagline":{"@id":"schema:description","@language": "en"},
    "photo":{"@id":"schema:image","@type":"@id"}
  }
,
  "@id": "http://id.manufacturer.com/gtin/05011476100885",
  "@type": [ "TradeItem"],
  "ean13":"5011476100885",
  "productName": "Cheerios 600g",
  "tagline": "Deliciously crunchy Os, packed with 4 whole grains.",
  "photo": "http://www.manufacturer.com/images/Cheerios600g.jpg"
}
```

The above JSON-LD results in exactly the same RDF triples as the JSON-LD given earlier in this section, even though all of the local names used in the body are different. The reason it is equivalent to the earlier JSON-LD example is that the full predicate URIs defined in the context section are the same.

3.4 Procedure for retailers to construct a simple block of JSON-LD to represent basic facts about any product offering, using the schema.org vocabulary

A retailer wishes to include create linked data about a product offering so that they can assert facts about that offering such as the offering price and a retailer's own product image, and enable others to link to this information. At the same time, the retailer wishes its information to be linked to the manufacturer's information about the same product.

3.4.1 Pre-Requisite

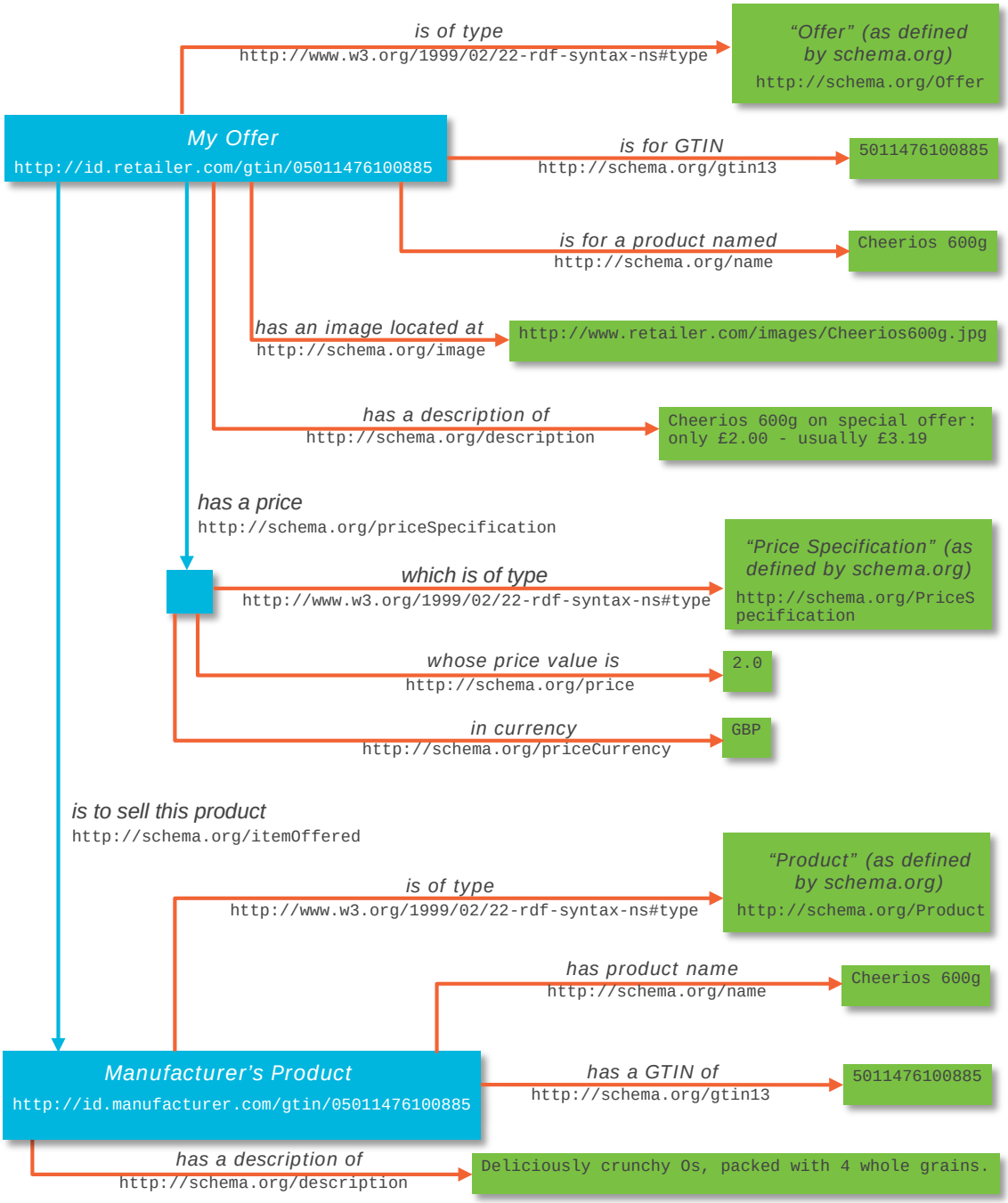
Sections 3.1, 3.2, and 3.3.

3.4.2 When Would I Use This?

Use this procedure when creating linked data for your product offering using the schema.org vocabulary.

3.4.3 How To?

Start with a visualisation of the facts we want to write:



In this figure, concepts are written in italics, and the URI representation of those concepts as used in RDF written below that.

This is rather more complicated than the previous example in Section 3.3 because the retailer needs to assert facts about their offer for a product (such as price information) – but they might also want to include facts asserted by the brand owner or manufacturer. Notice how the Offer has predicates that relate it to two other objects: one, the manufacturer's product which itself is the subject of its own descriptive triples; and two, the price which as a structured value is represented as a subject, with triples providing the price and currency as separate data values. Because the price structure only has local meaning within this triple graph, it does not need a globally unique URI of its own.

The `http://schema.org/Offer` contains the facts asserted by the retailer (e.g. about price information etc.), while the `http://schema.org/Product` may contains facts originally asserted by

the brand owner or manufacturer (e.g. about the product characteristics and specifications, as well as description).

The corresponding RDF triples we want to assert in this example are the following:

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix schema: <http://schema.org/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://id.retailer.com/gtin/05011476100885> rdf:type schema:Offer .
<http://id.retailer.com/gtin/05011476100885> schema:gtin13 "5011476100885" .
<http://id.retailer.com/gtin/05011476100885> schema:name "Cheerios 600g"@en .
<http://id.retailer.com/gtin/05011476100885> schema:description "Cheerios 600g on
special offer: only £2.00 - usually £3.19"@en .
<http://id.retailer.com/gtin/05011476100885> schema:image
<http://www.retailer.com/img/Cheerios-600g.jpg> .
<http://id.retailer.com/gtin/05011476100885> schema:priceSpecification _:b0 .
_:b0 rdf:type schema:PriceSpecification .
_:b0 schema:price "2.00"^^xsd:float .
_:b0 schema:priceCurrency "GBP" .
_:b0 schema:priceType "List" .

<http://id.retailer.com/gtin/05011476100885> schema:itemOffered
<http://id.manufacturer.com/gtin/05011476100885> .

<http://id.manufacturer.com/gtin/05011476100885> rdf:type schema:Product .
<http://id.manufacturer.com/gtin/05011476100885> schema:gtin13 "5011476100885" .
<http://id.manufacturer.com/gtin/05011476100885> schema.org:name "Cheerios 600g"@en
.
<http://id.manufacturer.com/gtin/05011476100885> schema:description "Deliciously
crunchy Os, packed with 4 whole grains"@en .
```

In RDF Turtle notation, the underscore before the colon in `_:b0` indicates that this is just a local name that has no significance outside this triple graph (corresponding to the blue square in the figure above).

The JSON-LD block looks like:

```
{
 "@context": {
   "schema": "http://schema.org/",
   "xsd": "http://www.w3.org/2001/XMLSchema#",

   "Offer":"schema:Offer",
   "Product":"schema:Product",

   "productName":{"@id":"schema:name","@language": "en"},
   "offerName":{"@id":"schema:name","@language": "en"},
   "productDescription":{"@id":"schema:description","@language": "en"},
   "offerDescription":{"@id":"schema:description","@language": "en"},
   "gtin13": {"@id":"schema:gtin13", "@type": "xsd:string" },

   "image":{"@id":"schema:image","@type":"@id"},
   "price":{"@id":"schema:price","@type":"xsd:float"},
   "currencyUnit":{"@id":"schema:priceCurrency","@type":"xsd:string"},
   "priceType":{"@id":"schema:priceType","@type":"xsd:string"},
   "hasPrice":{"@id":"schema:priceSpecification","@type":"@id"},
   "includes":{"@id":"schema:itemOffered","@type":"@id"}

  },
  "@id": "http://id.retailer.com/gtin/05011476100885",
  "@type": "Offer",
  "gtin13":"5011476100885",
  "offerName": "Cheerios 600g",
```

```
  "offerDescription": "Cheerios 600g on special offer: only £2.00 - usually
£3.19",
  "image":"http://www.retailer.com/img/Cheerios-600g.jpg",
  "hasPrice": {
    "price": "2.00",
    "currencyUnit": "GBP",
    "priceType": "List",
    "@type":"schema:PriceSpecification"
  },
  "includes": {
     "@id": "http://id.manufacturer.com/gtin/05011476100885",
     "@type": [ "Product" ],
     "gtin13":"5011476100885",
     "productName": "Cheerios 600g",
     "productDescription": "Deliciously crunchy Os, packed with 4 whole
grains."
  }
}
```

In RDF Turtle, the special notation `_:b0` had to be used to represent the local subject used for the price. In JSON-LD, this is expressed more naturally by simply nesting the price attributes within the value for `hasPrice`, thereby avoiding the need to introduce the name `_:b0`.

3.5 Procedure for brand owners or manufacturers to construct a simple block of JSON-LD to represent basic facts about any product, using the GS1 web vocabulary

A brand owner or manufacturer wishes to include create linked data about their product so that they can assert facts about that product such as technical specifications, ingredients, and nutritional information, and enable others (e.g. retailers) to link to this information. This is similar to the procedure in Section 3.3, but in this case we are using both schema.org vocabulary and the GS1 Web Vocabulary. This allows for the inclusion of a much richer set of product attributes.

This example shows a food product and nutritional attributes, but the GS1 Web Vocabulary includes specialised product attributes for many other product categories as well.

3.5.1 Pre-Requisite

Section 3.3.

3.5.2 When Would I Use This?

Use this procedure when creating linked data for your product using the extended GS1 Web Vocabulary offered by GS1.

3.5.3 How To?

Here is some sample JSON-LD:

```
{
 "@context": {
   "gs1": "http://gs1.org/voc/",
   "schema": "http://schema.org/",
   "xsd": "http://www.w3.org/2001/XMLSchema#",

   "TradeItem":"schema:Product",

   "tradeItemDescription":{"@id":"schema:description","@language": "en"},
   "gtin13": {"@id":"schema:gtin13", "@type": "xsd:string" },

   "image":{"@id":"schema:image","@type":"@id"},
```

```
   "healthClaimDescription":{"@id":"gs1:healthClaimDescription","@language": "en"},
   "allergenStatement":{"@id":"gs1:allergenStatement","@language": "en"},

   "gs1:measurementUnitCode": { "@type": "xsd:string" },
   "value":{"@id":"gs1:measurementValue","@type":"xsd:float"},
   "unit":{"@id":"gs1:measurementUnitCode","@type":"xsd:string"},

   "ingredientpercentage":{"@id":"gs1:ingredientContentPercentage","@type":"xsd:float"},
   "ingredientseq":{"@id":"gs1:ingredientSequence","@type":"xsd:integer"},
   "ingredientname":{"@id":"gs1:ingredientName","@language":"en"},

   "hasAllergenRelatedInformation":{"@id":"gs1:hasAllergenRelatedInformation","@type":"@id"},
   "hasIngredients":{"@id":"gs1:hasIngredients","@type":"@id"},
   "hasIngredientDetail":{"@id":"gs1:hasIngredientDetail","@type":"@id"},

   "nutrientBasisQuantity":{"@id":"gs1:nutrientBasisQuantity","@type":"@id"},
   "energyPerNutrientBasis":{"@id":"gs1:energyPerNutrientBasis","@type":"@id"},
   "proteinPerNutrientBasis":{"@id":"gs1:proteinPerNutrientBasis","@type":"@id"},
   "carbohydratesPerNutrientBasis":{"@id":"gs1:carbohydratesPerNutrientBasis","@type":"@id"},
   "sugarsPerNutrientBasis":{"@id":"gs1:sugarsPerNutrientBasis","@type":"@id"},
   "fatPerNutrientBasis":{"@id":"gs1:fatPerNutrientBasis","@type":"@id"},
   "saturatedFatPerNutrientBasis":{"@id":"gs1:saturatedFatPerNutrientBasis","@type":"@id"},
   "fibrePerNutrientBasis":{"@id":"gs1:fibrePerNutrientBasis","@type":"@id"},
   "sodiumPerNutrientBasis":{"@id":"gs1:sodiumPerNutrientBasis","@type":"@id"},
   "saltPerNutrientBasis":{"@id":"gs1:saltPerNutrientBasis","@type":"@id"},
   "vitaminCPerNutrientBasis":{"@id":"gs1:vitaminCPerNutrientBasis","@type":"@id"},
   "thiaminPerNutrientBasis":{"@id":"gs1:thiaminPerNutrientBasis","@type":"@id"},
   "riboflavinPerNutrientBasis":{"@id":"gs1:riboflavinPerNutrientBasis","@type":"@id"},
   "niacinPerNutrientBasis":{"@id":"gs1:niacinPerNutrientBasis","@type":"@id"},
   "vitaminB6PerNutrientBasis":{"@id":"gs1:vitaminB6PerNutrientBasis","@type":"@id"},
   "folicAcidPerNutrientBasis":{"@id":"gs1:folicAcidPerNutrientBasis","@type":"@id"},
   "vitaminB12PerNutrientBasis":{"@id":"gs1:vitaminB12PerNutrientBasis","@type":"@id"},

"pantothenicAcidPerNutrientBasis":{"@id":"gs1:pantothenicAcidPerNutrientBasis","@type":"@id"},
   "calciumPerNutrientBasis":{"@id":"gs1:calciumPerNutrientBasis","@type":"@id"},
   "ironPerNutrientBasis":{"@id":"gs1:ironPerNutrientBasis","@type":"@id"},

   "dv":{"@id":"gs1:dailyValueIntakePercent","@type":"xsd:float"},

   "Ingredient":"gs1:FoodAndBeverageIngredientDetail",
   "Measurement":"gs1:NutritionMeasurementType"

  },
  "@id": "http://id.manufacturer.com/gtin/05011476100885",
  "gtin13":"5011476100885",
  "@type": [ "TradeItem"],
  "tradeItemDescription": "Deliciously crunchy Os, packed with 4 whole grains. Say Yes to
Cheerios",
  "healthClaimDescription":"8 Vitamins & Iron, Source of Calcium & High in Fibre",
  "hasAllergenRelatedInformation": {"@type":
"gs1:AllergenRelatedInformation","allergenStatement":"May contain nut traces"},
  "hasIngredients": {"@type": "gs1:FoodAndBeverageIngredient","hasIngredientDetail":[
{"@type":"Ingredient","ingredientseq":"1","ingredientname":"Cereal
Grains","ingredientpercentage":"77.5"},
{"@type":"Ingredient","ingredientseq":"2","ingredientname":"Whole Grain
OATS","ingredientpercentage":"38.0"},
{"@type":"Ingredient","ingredientseq":"3","ingredientname":"Whole Grain
WHEAT","ingredientpercentage":"18.6"},
{"@type":"Ingredient","ingredientseq":"4","ingredientname":"Whole Grain
BARLEY","ingredientpercentage":"12.8"},
{"@type":"Ingredient","ingredientseq":"5","ingredientname":"Whole Grain
Rice","ingredientpercentage":"5.5"},
{"@type":"Ingredient","ingredientseq":"6","ingredientname":"Whole Grain
Maize","ingredientpercentage":"2.6"},
{"@type":"Ingredient","ingredientseq":"7","ingredientname":"Sugar"},
{"@type":"Ingredient","ingredientseq":"8","ingredientname":"Wheat Starch"},
{"@type":"Ingredient","ingredientseq":"9","ingredientname":"Partially Inverted Brown Sugar
Syrup"},
{"@type":"Ingredient","ingredientseq":"10","ingredientname":"Salt"},
{"@type":"Ingredient","ingredientseq":"11","ingredientname":"Tripotassium Phosphate"},
{"@type":"Ingredient","ingredientseq":"12","ingredientname":"Sunflower Oil"},
{"@type":"Ingredient","ingredientseq":"13","ingredientname":"Colours: Caramel, Annatto,
Carotene"},
{"@type":"Ingredient","ingredientseq":"14","ingredientname":"Antioxidant: Tocopherals"},
{"@type":"Ingredient","ingredientseq":"15","ingredientname":"Vitamin C"},
```

```
{"@type":"Ingredient","ingredientseq":"16","ingredientname":"Niacin"},
{"@type":"Ingredient","ingredientseq":"17","ingredientname":"Pantothenic Acid"},
{"@type":"Ingredient","ingredientseq":"18","ingredientname":"Thiamin (B1)"},
{"@type":"Ingredient","ingredientseq":"19","ingredientname":"Vitamin B6"},
{"@type":"Ingredient","ingredientseq":"20","ingredientname":"Riboflavin (B2)"},
{"@type":"Ingredient","ingredientseq":"21","ingredientname":"Folic Acid (Folacin)"},
{"@type":"Ingredient","ingredientseq":"22","ingredientname":"Vitamin B12"},
{"@type":"Ingredient","ingredientseq":"23","ingredientname":"Calcium Carbonate"},
{"@type":"Ingredient","ingredientseq":"24","ingredientname":"Iron"}
]},
  "nutrientBasisQuantity":{"@type":"Measurement","value":"100","unit":"GRM"},
  "energyPerNutrientBasis":
[{"@type":"Measurement","value":"1615","unit":"KJO"},{"@type":"Measurement","value":"382","unit
":"E14"}],
  "proteinPerNutrientBasis":{"@type":"Measurement","value":"8.6","unit":"GRM"},
  "carbohydratesPerNutrientBasis":{"@type":"Measurement","value":"74.3","unit":"GRM"},
  "sugarsPerNutrientBasis":{"@type":"Measurement","value":"21.4","unit":"GRM"},
  "fatPerNutrientBasis":{"@type":"Measurement","value":"4.0","unit":"GRM"},
  "saturatedFatPerNutrientBasis":{"@type":"Measurement","value":"1.0","unit":"GRM"},
  "fibrePerNutrientBasis":{"@type":"Measurement","value":"7.1","unit":"GRM"},
  "sodiumPerNutrientBasis":{"@type":"Measurement","value":"0.41","unit":"GRM"},
  "saltPerNutrientBasis":{"@type":"Measurement","value":"1.04","unit":"GRM"},
  "vitaminCPerNutrientBasis":{"@type":"Measurement","value":"71.0","unit":"MGM","dv":"89"},
  "thiaminPerNutrientBasis":{"@type":"Measurement","value":"1.24","unit":"MGM","dv":"113"},
  "riboflavinPerNutrientBasis":{"@type":"Measurement","value":"1.10","unit":"MGM","dv":"79"},
  niacinPerNutrientBasis":{"@type":"Measurement","value":"14.0","unit":"MGM","dv":"88"},
   "vitaminB6PerNutrientBasis":{"@type":"Measurement","value":"1.20","unit":"MGM","dv":"86"},
  "folicAcidPerNutrientBasis":{"@type":"Measurement","value":"200","unit":"MC","dv":"100"},
  "vitaminB12PerNutrientBasis":{"@type":"Measurement","value":"1.90","unit":"MC","dv":"76"},

"pantothenicAcidPerNutrientBasis":{"@type":"Measurement","value":"4.40","unit":"MGM","dv":"73"}
,
  "calciumPerNutrientBasis":{"@type":"Measurement","value":"460","unit":"MGM","dv":"58"},
  "ironPerNutrientBasis":{"@type":"Measurement","value":"14.7","unit":"MGM","dv":"105"}

}
```


Explanation:

The context section (shaded orange) references three namespaces – the GS1 vocabulary, the schema.org vocabulary and XSD (XML Schema Definition). (XSD is used for standard data types such as `xsd:float` and `xsd:integer`). The RDF namespace is implicitly included through the JSON-LD `@type` keyword, which maps to `rdf:type`.

Some basic fields such as the description of the offer or the trade item (product), the `gtin13` property, image and price information are mapped to terms from the schema.org vocabulary.

Specialised terms specific to food and beverage products are mapped to terms from the gs1 vocabulary.

Some of these specialised terms for food product ingredients or nutritional information do not take simple string values but instead take complex data values such as a `gs1:NutritionMeasurementType` (which can be used to express a quantity, a unit of measure and percentage of the recommended daily intake of a nutrient as recommended by authorities of the target market) – or a `gs1:FoodAndBeverageIngredientDetail` (which can accept an ingredient sequence number, ingredient name and ingredient as a percentage of the total composition of the product).

Note: A***bout properties with multiple values, lists, sequences etc.***

Another important point to note is that unlike RDF Turtle or N-Triples, in JSON-LD, the name of each property or predicate **should appear only once** in the data block. There may be situations where in RDF triples we might write several triples each containing the same property or predicate, perhaps using blank nodes if the value is not a simple data type. When we want to express these in JSON-LD, we must write the name of the property or predicate **once only** – and use a list for the sets of values corresponding to that property. In the example above, we can see examples of lists in JSON-LD (enclosed in square brackets) for the properties 'hasIngredientDetail' and 'energyPerNutrientBasis'. Lists are used in these examples to allow for multiple ingredients and for two different energy units, respectively.

3.6 Procedure for retailers to construct a simple block of JSON-LD to represent basic facts about any product offering, using the GS1 web vocabulary

A retailer wishes to include create linked data about a product offering so that they can assert facts about that offering such as the offering price and a retailer's own product image, and enable others to link to this information. At the same time, the retailer also wishes to include detailed product information such as ingredients and nutritional information. This is similar to the procedure in Section 3.43.3, but in this case we are using both schema.org vocabulary and the GS1 Web Vocabulary. This allows for the inclusion of a much richer set of product attributes.

This example shows a food product and nutritional attributes, but the GS1 Web Vocabulary includes specialised product attributes for many other product categories as well.

3.6.1 Pre-Requisite

Sections 3.4.

3.6.2 When Would I Use This?

The following example shows how a retailer can use the GS1 vocabulary in combination with the schema.org vocabulary to write facts about a product offer for a food product. In this example, we have used schema.org properties and classes (shown in red) wherever we can express properties sufficiently precisely using the schema.org vocabulary. For the nutritional information and ingredients list, we use the GS1 vocabulary because it supports a wider variety of nutrients and also allows us to specify an explicit nutrient basis quantity (e.g. 100g or 100ml of product), so that there is no ambiguity about what the quantities (e.g. protein content) relate to.

However, we note that schema.org does define some related properties in *http://schema.org/NutritionInformation* and support expression of a list of ingredients within the context of a *http://schema.org/Recipe* - but schema.org does not currently provide any guidance about how these might be applied reliably to express the nutritional information or ingredients of a food product.

3.6.3 How To?

Here is some sample JSON-LD:

```
{
 "@context": {
   "gs1": "http://gs1.org/voc/",
   "schema": "http://schema.org/",
   "xsd": "http://www.w3.org/2001/XMLSchema#",

   "TradeItem":"schema:Product",
   "Offering":"schema:Offer",

   "offerDescription":{"@id":"schema:description","@language": "en"},
   "tradeItemDescription":{"@id":"schema:description","@language": "en"},
   "gtin13": {"@id":"schema:gtin13", "@type": "xsd:string" },

   "image":{"@id":"schema:image","@type":"@id"},
   "price":{"@id":"schema:price","@type":"xsd:float"},
   "currencyUnit":{"@id":"schema:priceCurrency","@type":"xsd:string"},
   "priceType":{"@id":"schema:priceType","@type":"xsd:string"},
   "hasPrice":{"@id":"schema:priceSpecification","@type":"@id"},
   "includes":{"@id":"schema:itemOffered","@type":"@id"},

   "healthClaimDescription":{"@id":"gs1:healthClaimDescription","@language": "en"},
   "allergenStatement":{"@id":"gs1:allergenStatement","@language": "en"},

   "gs1:measurementUnitCode": { "@type": "xsd:string" },
   "value":{"@id":"gs1:measurementValue","@type":"xsd:float"},
   "unit":{"@id":"gs1:measurementUnitCode","@type":"xsd:string"},

   "ingredientpercentage":{"@id":"gs1:ingredientContentPercentage","@type":"xsd:float"},
   "ingredientseq":{"@id":"gs1:ingredientSequence","@type":"xsd:integer"},
```

```
   "ingredientname":{"@id":"gs1:ingredientName","@language":"en"},

   "hasAllergenRelatedInformation":{"@id":"gs1:hasAllergenRelatedInformation","@type":"@id"},
   "hasIngredients":{"@id":"gs1:hasIngredients","@type":"@id"},
   "hasIngredientDetail":{"@id":"gs1:hasIngredientDetail","@type":"@id"},

   "nutrientBasisQuantity":{"@id":"gs1:nutrientBasisQuantity","@type":"@id"},
   "energyPerNutrientBasis":{"@id":"gs1:energyPerNutrientBasis","@type":"@id"},
   "proteinPerNutrientBasis":{"@id":"gs1:proteinPerNutrientBasis","@type":"@id"},
   "carbohydratesPerNutrientBasis":{"@id":"gs1:carbohydratesPerNutrientBasis","@type":"@id"},
   "sugarsPerNutrientBasis":{"@id":"gs1:sugarsPerNutrientBasis","@type":"@id"},
   "fatPerNutrientBasis":{"@id":"gs1:fatPerNutrientBasis","@type":"@id"},
   "saturatedFatPerNutrientBasis":{"@id":"gs1:saturatedFatPerNutrientBasis","@type":"@id"},
   "fibrePerNutrientBasis":{"@id":"gs1:fibrePerNutrientBasis","@type":"@id"},
   "sodiumPerNutrientBasis":{"@id":"gs1:sodiumPerNutrientBasis","@type":"@id"},
   "saltPerNutrientBasis":{"@id":"gs1:saltPerNutrientBasis","@type":"@id"},
   "vitaminCPerNutrientBasis":{"@id":"gs1:vitaminCPerNutrientBasis","@type":"@id"},
   "thiaminPerNutrientBasis":{"@id":"gs1:thiaminPerNutrientBasis","@type":"@id"},
   "riboflavinPerNutrientBasis":{"@id":"gs1:riboflavinPerNutrientBasis","@type":"@id"},
   "niacinPerNutrientBasis":{"@id":"gs1:niacinPerNutrientBasis","@type":"@id"},
   "vitaminB6PerNutrientBasis":{"@id":"gs1:vitaminB6PerNutrientBasis","@type":"@id"},
   "folicAcidPerNutrientBasis":{"@id":"gs1:folicAcidPerNutrientBasis","@type":"@id"},
   "vitaminB12PerNutrientBasis":{"@id":"gs1:vitaminB12PerNutrientBasis","@type":"@id"},

"pantothenicAcidPerNutrientBasis":{"@id":"gs1:pantothenicAcidPerNutrientBasis","@type":"@id"},
   "calciumPerNutrientBasis":{"@id":"gs1:calciumPerNutrientBasis","@type":"@id"},
   "ironPerNutrientBasis":{"@id":"gs1:ironPerNutrientBasis","@type":"@id"},

   "dv":{"@id":"gs1:dailyValueIntakePercent","@type":"xsd:float"},

   "Ingredient":"gs1:FoodAndBeverageIngredientDetail",
   "Measurement":"gs1:NutritionMeasurementType"

  },
 "@id": "http://id.retailer.com/gtin/05011476100885",
 "@type": "Offering",
 "gtin13":"5011476100885",
 "offerDescription": "Nestle Cheerios Cereal 600G",
 "image":"http://www.retailer.com/Groceries/pi/885/5011476100885/IDShot_225x225.jpg",
 "hasPrice": {
   "price": "2.00",
   "currencyUnit": "GBP",
   "priceType": "List",
   "@type":"schema:PriceSpecification"
 },
 "includes": {
   "@id": "http://id.manufacturer.com/gtin/05011476100885",
   "gtin13":"5011476100885",
   "@type": [ "TradeItem"],
   "tradeItemDescription": "Deliciously crunchy Os, packed with 4 whole grains. Say Yes to
Cheerios",
   "healthClaimDescription":"8 Vitamins & Iron, Source of Calcium & High in Fibre",
   "hasAllergenRelatedInformation": {"@type":
"gs1:AllergenRelatedInformation","allergenStatement":"May contain nut traces"},
   "hasIngredients": {"@type": "gs1:FoodAndBeverageIngredient","hasIngredientDetail":[
{"@type":"Ingredient","ingredientseq":"1","ingredientname":"Cereal
Grains","ingredientpercentage":"77.5"},
{"@type":"Ingredient","ingredientseq":"2","ingredientname":"Whole Grain
OATS","ingredientpercentage":"38.0"},
{"@type":"Ingredient","ingredientseq":"3","ingredientname":"Whole Grain
WHEAT","ingredientpercentage":"18.6"},
{"@type":"Ingredient","ingredientseq":"4","ingredientname":"Whole Grain
BARLEY","ingredientpercentage":"12.8"},
{"@type":"Ingredient","ingredientseq":"5","ingredientname":"Whole Grain
Rice","ingredientpercentage":"5.5"},
{"@type":"Ingredient","ingredientseq":"6","ingredientname":"Whole Grain
Maize","ingredientpercentage":"2.6"},
{"@type":"Ingredient","ingredientseq":"7","ingredientname":"Sugar"},
{"@type":"Ingredient","ingredientseq":"8","ingredientname":"Wheat Starch"},
{"@type":"Ingredient","ingredientseq":"9","ingredientname":"Partially Inverted Brown Sugar
Syrup"},
{"@type":"Ingredient","ingredientseq":"10","ingredientname":"Salt"},
{"@type":"Ingredient","ingredientseq":"11","ingredientname":"Tripotassium Phosphate"},
{"@type":"Ingredient","ingredientseq":"12","ingredientname":"Sunflower Oil"},
```



```
{"@type":"Ingredient","ingredientseq":"13","ingredientname":"Colours: Caramel, Annatto,
Carotene"},
{"@type":"Ingredient","ingredientseq":"14","ingredientname":"Antioxidant: Tocopherals"},
{"@type":"Ingredient","ingredientseq":"15","ingredientname":"Vitamin C"},
{"@type":"Ingredient","ingredientseq":"16","ingredientname":"Niacin"},
{"@type":"Ingredient","ingredientseq":"17","ingredientname":"Pantothenic Acid"},
{"@type":"Ingredient","ingredientseq":"18","ingredientname":"Thiamin (B1)"},
{"@type":"Ingredient","ingredientseq":"19","ingredientname":"Vitamin B6"},
{"@type":"Ingredient","ingredientseq":"20","ingredientname":"Riboflavin (B2)"},
{"@type":"Ingredient","ingredientseq":"21","ingredientname":"Folic Acid (Folacin)"},
{"@type":"Ingredient","ingredientseq":"22","ingredientname":"Vitamin B12"},
{"@type":"Ingredient","ingredientseq":"23","ingredientname":"Calcium Carbonate"},
{"@type":"Ingredient","ingredientseq":"24","ingredientname":"Iron"}
]},
   "nutrientBasisQuantity":{"@type":"Measurement","value":"100","unit":"GRM"},
   "energyPerNutrientBasis":
[{"@type":"Measurement","value":"1615","unit":"KJO"},{"@type":"Measurement","value":"382","unit
":"E14"}],
   "proteinPerNutrientBasis":{"@type":"Measurement","value":"8.6","unit":"GRM"},
   "carbohydratesPerNutrientBasis":{"@type":"Measurement","value":"74.3","unit":"GRM"},
   "sugarsPerNutrientBasis":{"@type":"Measurement","value":"21.4","unit":"GRM"},
   "fatPerNutrientBasis":{"@type":"Measurement","value":"4.0","unit":"GRM"},
   "saturatedFatPerNutrientBasis":{"@type":"Measurement","value":"1.0","unit":"GRM"},
   "fibrePerNutrientBasis":{"@type":"Measurement","value":"7.1","unit":"GRM"},
   "sodiumPerNutrientBasis":{"@type":"Measurement","value":"0.41","unit":"GRM"},
   "saltPerNutrientBasis":{"@type":"Measurement","value":"1.04","unit":"GRM"},
   "vitaminCPerNutrientBasis":{"@type":"Measurement","value":"71.0","unit":"MGM","dv":"89"},
   "thiaminPerNutrientBasis":{"@type":"Measurement","value":"1.24","unit":"MGM","dv":"113"},
   "riboflavinPerNutrientBasis":{"@type":"Measurement","value":"1.10","unit":"MGM","dv":"79"},
   "niacinPerNutrientBasis":{"@type":"Measurement","value":"14.0","unit":"MGM","dv":"88"},
   "vitaminB6PerNutrientBasis":{"@type":"Measurement","value":"1.20","unit":"MGM","dv":"86"},
   "folicAcidPerNutrientBasis":{"@type":"Measurement","value":"200","unit":"MC","dv":"100"},
   "vitaminB12PerNutrientBasis":{"@type":"Measurement","value":"1.90","unit":"MC","dv":"76"},

"pantothenicAcidPerNutrientBasis":{"@type":"Measurement","value":"4.40","unit":"MGM","dv":"73"}
,
   "calciumPerNutrientBasis":{"@type":"Measurement","value":"460","unit":"MGM","dv":"58"},
   "ironPerNutrientBasis":{"@type":"Measurement","value":"14.7","unit":"MGM","dv":"105"}
 }
}
```


Explanation:

The context section (shaded orange) references three namespaces – the GS1 vocabulary, the schema.org vocabulary and XSD (XML Schema Definition). (XSD is used for standard data types such as `xsd:float` and `xsd:integer`). The RDF namespace is implicitly included through the JSON-LD `@type` keyword, which maps to `rdf:type`.

Some basic fields such as the description of the offer or the trade item (product), the `gtin13` property, image and price information are mapped to terms from the schema.org vocabulary.

Specialised terms specific to food and beverage products are mapped to terms from the `gs1` vocabulary.

Some of these specialised terms for food product ingredients or nutritional information do not take simple string values but instead take complex data values such as a `gs1:NutritionMeasurementType` (which can be used to express a quantity, a unit of measure and percentage of the recommended daily intake of a nutrient as recommended by authorities of the target market) – or a `gs1:FoodAndBeverageIngredientDetail` (which can accept an ingredient sequence number, ingredient name and ingredient as a percentage of the total composition of the product).

Note: ***About properties with multiple values, lists, sequences etc.***

Another important point to note is that unlike RDF Turtle or N-Triples, in JSON-LD, the name of each property or predicate **should appear only once** in the data block. There may be situations where in RDF triples we might write several triples each containing the same property or predicate, perhaps using blank nodes if the value is not a simple data type. When we want to express these in JSON-LD, we must write the name of the property or predicate

once only – and use a list for the sets of values corresponding to that property. In the example above, we can see examples of lists in JSON-LD (enclosed in square brackets) for the properties 'hasIngredientDetail' and 'energyPerNutrientBasis'. Lists are used in these examples to allow for multiple ingredients and for two different energy units, respectively.

3.7 Procedure for serving a block of JSON-LD via an existing web page, using embedding – one product per page

The publisher of a web page includes a block of JSON-LD that describes the single product appearing on that page.

3.7.1 Pre-Requisite

Section 3.3, 3.4, 3.5, or 3.6.

3.7.2 When Would I Use This?

Use this procedure when there is a single product described on a given web page, and you want to include structured data about that product.

3.7.3 How To?

You add JSON-LD to a web page simply by putting it inside of a `<script>` tag that specifies an Internet Media type of `application/ld+json`. This can be inserted within the <head> section of your page, like this:

```
<html>
      <head>
            <script type="application/ld+json">
                  (JSON-LD block goes here)
            </script>
            ... (rest of head section)
      </head>
      <body>
            ... (visible part of the web page)
      </body>
</html>
```

Alternatively, the JSON-LD can be added as the last child element within the <body> section of your page, like this:

```
<html>
      <head>
            ... (rest of head section)
      </head>
      <body>
            ... (visible part of the web page)
            <script type="application/ld+json">
                  (JSON-LD block goes here)
            </script>
      </body>
</html>
```

Either way, the JSON-LD block will be understood as referring to the entire page.

This illustrates the chief advantage of JSON-LD compared to RDFa or other means of embedding structured data in a web page: unlike inline formats such as RDFa and Microdata, JSON-LD is inserted in just one place in the page markup, well away from the visible content. This makes adding JSON-LD to a web page much easier and much less prone to error.

It is important to note that when JSON-LD (or any other structured data format) is added to a web page, the semantic information in machine readable format must match the information in the human readable section. Failure to do so is considered abusive use by search engines, which could result in a lower rank for the web page or the page not being listed at all.

3.8 Procedure for serving a block of JSON-LD via an existing web page, using embedding – procedure for multiple products per page

The publisher of a web page includes a block of JSON-LD that describes multiple products appearing on that page.

3.8.1 Pre-Requisite

Section 3.3, 3.4, 3.5, or 3.6.

3.8.2 When Would I Use This?

Use this procedure when there are more than one single product described on a given web page, and you want to include structured data about each of those products.

3.8.3 How To?

The procedure is almost the same as presented in Section 3.7, except that separate JSON-LD blocks, each within their own `<script>` tags, are included for each product. Each JSON-LD block corresponds to exactly one GTIN, and contains one subject for the Product GTIN and at most one subject for the Offer GTIN (the latter only being applicable for a retailer's web page).

As in Section 3.7, you add JSON-LD to a web page by putting it inside of a `<script>` tag that specifies an Internet Media type of `application/ld+json`. This can be inserted within the <head> section of your page, like this:

```
<html>
      <head>
            <script type="application/ld+json">
                  (JSON-LD block for GTIN 1 goes here)
            </script>
            <script type="application/ld+json">
                  (JSON-LD block for GTIN 2 goes here)
            </script>
            ... (and so forth for remaining GTINs)
            ... (rest of head section)
      </head>
      <body>
            ... (visible part of the web page)
      </body>
</html>
```

Alternatively, the JSON-LD blocks can be added as the last child element within the <body> section of your page, like this:

```
<html>
      <head>
            ... (rest of head section)
      </head>
      <body>
            ... (visible part of the web page)
            <script type="application/ld+json">
                  (JSON-LD block for GTIN 1 goes here)
            </script>
            <script type="application/ld+json">
                  (JSON-LD block for GTIN 2 goes here)
            </script>
            ... (and so forth for remaining GTINs)
      </body>
</html>
```

It is important to note that when JSON-LD (or any other structured data format) is added to a web page, the semantic information in machine readable format must match the information in the human readable section. Failure to do so is considered abusive use by search engines, which could result in a lower rank for the web page or the page not being listed at all.

In the case of multiple products per web page, there should be exactly as many JSON-LD blocks as there are GTINs in the human-readable portion of the page, and the JSON-LD blocks should appear in the same order as the order the corresponding GTINs appear in the HTML markup for the human-readable portion. To further establish the correspondence between the JSON-LD and the human-readable portion, and because of policies regarding abuse as discussed above, all attributes in the JSON-LD should match information presented in the human-readable HTML (a notable exception being the GTIN itself, which might not appear in the human-readable section).

3.9 Procedure for serving a standalone block of JSON-LD in isolation via a webserver

The data publisher exposes web resources that return a JSON-LD representation of the resource when dereferenced (as opposed to an HTML web page that embeds the JSON-LD).

3.9.1 Pre-Requisite

Section 3.3, 3.4, 3.5, or 3.6.

3.9.2 When Would I Use This?

Many modern front-end frameworks, such as *AngularJS*, use JavaScript to manipulate the webpage and asynchronously load JSON data from an API. Using JSON-LD rather than plain-old JSON allows the use of shared identifiers for properties and the possibility of embedding links to other resources into the JSON (regular JSON does not support the URI datatype). Using this approach to make the data and external resource that can be referenced to makes it possible to share and re-use data across different webpages without having to embed the same data into each and every page. This can improve cacheability of resources and reduce the tidal wave effect whereby a small change can result in many hundreds or thousands of HTML pages needing to be updated.

3.9.3 How To?

JSON-LD data or context files can be served using a conventional webserver. However, it is important to configure the webserver to specify the appropriate MIME type in the Header information before it sends the JSON-LD file. The MIME type for JSON-LD is application/ld+json

If using an Apache webserver, you can achieve this by modifying the .htaccess file in the same directory as the JSON-LD files so that it includes the following lines:

```
Header set Access-Control-Allow-Origin "*"
AddType application/ld+json .jsonld
```

The first line enables Cross-Origin Resource Sharing (CORS) [see *http://enable-cors.org/*], so that javascript from other domains can access your JSON-LD files.

The second line forces the webserver to indicate a MIME type of `application/ld+json` whenever it serves a JSON-LD file, provided that the JSON-LD files are named with a `.jsonld` filename suffix.

3.10 Procedure for checking that structured data is correctly formatted

Once you have created JSON-LD for your product or product offering you will want to check that it is formatted correctly so that it can be processed by any applications and apps that may wish to consume it. This section explains the procedure and tools to help you achieve this.

3.10.1 Pre-Requisite

Section 3.7 or 3.8.

3.10.2 When Would I Use This?

Use one of the tools suggested to check that your data is syntactically correct and that it will be interpreted in the manner expected.

3.10.3 How To?

The JSON-LD Playground tool at http://json-ld.org/playground/index.html can be used to check the JSON-LD you have generated.

You can use the JSON-LD contained at the following web address to see results.

http://www.autoidlabs.org.uk/GS1Digital/Demos/GS1vocab/gs1JSON-LD-Demo.html

Just view the page source and paste the JSON-LD block in to the JSON-LD Playground form. (You must exclude the enclosing `<script>` tags as these are not part of the JSON-LD block.) The tool will check and report upon any syntax errors (e.g., "JSON markup - SyntaxError: Unexpected token {"). You can also use the tool to view your content in a number of different formats to help ensure the intended meaning of your JSON-LD.

Another useful tool can be found at http://linter.structured-data.org/. By pasting your page URL or uploading your page content you can use this tool to get a visual confirmation of the structured data in your page.

3.11 Procedure for accessing structured data in a JSON-LD block using JavaScript within the same web page

The publisher of a web page wishes to exploit the embedded JSON-LD content for other purposes within the web page itself.

3.11.1 Pre-Requisite

Section 3.7 or 3.8.

3.11.2 When Would I Use This?

Modern web pages do many data manipulations in Javascript. Embedding the information once in JSON-LD and then using it from Javascript can be useful for the following use-cases:

- Building rich user interfaces: instead of having duplicate content in the HTML portion of the web page and the JSON-LD, the web page includes just the JSON-LD and Javascript code that reads the JSON-LD to populate the user-facing content (via the DOM). This can offer benefits including:
  - Pagination of long content.
  - Translation of attributes into icons or procedurally generated graphics.
  - Displaying information in various languages without requiring a page-reload.
- Populating tracking data. Systems like Google Analytics take their data in Javascript structures, which can be populated by reading the JSON-LD data. This enables tracking by the various attributes that are added.

3.11.3 How To?

View the instructions at:

http://www.autoidlabs.org.uk/GS1Digital/Demos/GS1vocab/gs1JSON-LD-with-JavaScript.html

This page uses the same block of JSON-LD as the previous example and explains how JavaScript can access the data.

Note that JavaScript is not natively aware of JSON-LD, which means that it ignores the `@context` header and does not expand local keys or *`prefix`*`:`*`name`* constucts to full URIs, nor is it aware of `@type` or `@language`.

It is possible to access the data from JSON-LD – but not always via the dot (.) notation familiar in JSON.

A Appendix: Technical background for deploying Linked Data about products

What is the semantic web?

According to the World Wide Web Consortium (W3C) [http://www.w3.org/2001/sw/], The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries. It is a collaborative effort led by W3C with participation from a large number of researchers and industrial partners. It is based on the Resource Description Framework (RDF). It refers to a collection of technologies that can be used to transform the web of documents (e.g. web pages) into a global web of interlinked interoperable data that is machine-interpretable because the meaning of each data relationship is explicitly stated – and because the semantic web uses HTTP URIs (e.g. web addresses), it is possible to access related data as well as definitions of properties and attributes, multi-lingual names and descriptions simply via a regular HTTP web request.

What is Linked Data?

According to LinkedData.org, Linked Data is about using the Web to connect related data that wasn't previously linked, or using the Web to lower the barriers to linking data currently linked using other methods. Linked Data is sometimes considered as being either synonymous with the Semantic Web or being a subset of it. Linked Data can be provided and retrieved via web requests, either as standalone data – or embedded within regular web pages, as additional semantic markup of the facts contained within the page, which are accessible without ambiguities to software including search engines, smartphone apps etc.

See Section 1.3 for a brief introduction to Linked Data.

How is the data structured?

In Semantic Web / Linked Data technology, we don't think of the data as being structured in well-defined tables with rows and columns as in a relational database. The Semantic Web uses a simpler data structure in which facts, factual claims or data relationships are expressed as a directed graph of data. You can think of a graph of data as being very similar to a mind-map. A mind-map uses circles, ovals or rectangles to represent 'things' and arrows connecting these 'things' to represent the relationships between them.

In order to convert this 'mind-map' or 'graph' of data relationships from a pictorial representation to a format that can be processed by computer software, we usually represent each arrow on the mind-map as a triple that connects a subject (the 'thing' being described, at the start of the arrow) to an object (another 'thing' that appears at the end of the arrow). The arrow itself corresponds to a specific named property or predicate, which represents the data relationship that connects the subject to the object.

In this way, even very complicated data structures can be collapsed to essentially a 3-column table of 'triples'. This is the essence of Resource Description Framework (RDF) - a W3C technical standard that is at the foundation of the Semantic Web technology stack. There are a number of ways in which such RDF data can be exchanged or communicated. These include inline markup formats such as RDFa (RDF in annotations) or Microdata, and block-oriented formats such as JSON-LD (JavaScript Object Notation for Linked Data).

RDF Triples - Subject, Predicate, Object

As mentioned above, RDF enables us to write simple logical sentences to express factual assertions (e.g. a product has a specific weight) in a way that computer software can use, in order to 'understand' the meaning and potentially even generate some new facts ('inferencing') from existing facts that are explicitly stated, either by making use of precise logical assertions defined in an ontology - or by using user-defined rules in a query language such as SPARQL (see the SPARQL CONSTRUCT mechanism).

Use URIs instead of words

When we write facts in RDF, instead of using simple text string or words to identify things and relationships, we use HTTP URIs where possible. The exception to this is for simple literal values

such as numbers, dates or when we want to use a text string to provide a label, description or definition for something.

The advantage of using HTTP URIs is that they are globally unambiguous and can be created in a very decentralised manner. Anybody can create HTTP URIs by first obtaining an Internet domain name (or using one you already have, such as the domain of your website) and then using this in the "authority" portion of a URI. Because the domain name is unique, the HTTP URIs you create will not accidentally clash with HTTP URIs created by others.

While an HTTP URI does not have to be an actual web address to be usable in Linked Data, in practice it is helpful if an HTTP URI appearing in Linked Data can actually be used to make a web request ("dereferenced") that returns some useful information about the thing the HTTP URI represents. That way, if you want to find more information about a thing that is identified by an HTTP URI, or find the definition of a property or predicate identified by an HTTP URI, you can try making a web request for it.

Although you might be redirected to an alternative URI that delivers the information, you can typically expect to receive some useful information as a result of such a web request. This can include multi-lingual labels, descriptions and definitions, as well as links to other related things (also identified by HTTP URIs), and where the relationship of each link indicates a specific property or relationship.

What are vocabularies and what are ontologies?

Vocabularies and ontologies provide lists of concepts, classes (types of thing) and properties or predicates (relationships, attributes), together with their definitions. Examples of vocabularies include schema.org, GoodRelations, vCard, *Friend Of A Friend (FOAF)*, *Dublin Core* and the new GS1 vocabulary that is being developed in the GS1 SmartSearch work group.

Ontologies go a step further than vocabularies because they typically also include some very precise logical statements about the classes and properties that allow computer software to do some automated logical reasoning. For example, an ontology can make use of W3C technical standards such as *RDF Schema (RDFS)* and the *Web Ontology Language (OWL)* to make such statements. For example, we can define a father as being a sub-property of a parent. We can say that 'hasDateOfBirth' is only allowed to have one value for any specified thing. We can say that 'hasAncestor' is transitive, which means that if computer software sees a 'hasAncestor' relationship between you and one of your parents, and between one of your parents and one of your grandparents, it can use the transitive property to reason or infer that there is also a 'hasAncestor' property between you and your grandparents and your great-grandparents, etc. Classes can also be marked as being mutually disjoint (e.g. letters and digits are both subclasses of characters but have no overlapping members).

How is Linked Data published and made available by the publisher?

Linked Data can be embedded within existing web pages, either as inline markup using RDFa or Microdata or as a block of structured data, using JSON-LD markup. See Sections 3.7 and 3.8 for examples of how to embed a block of JSON-LD within an existing web page.

Linked Data can also be served directly, using a web server, provided that the appropriate *Internet Media Type (MIME)* headers are emitted before the data is served. See Section 3.9 for guidance about serving a block of JSON-LD directly, without embedding in a web page.

Another approach for serving Linked Data on the web is via the use of *SPARQL* endpoints. These provide an online query interface using the W3C SPARQL query protocol standard. In this situation, data need not be provided as a complete dump of Linked Data; instead the SPARQL endpoint can respond to SPARQL queries, perform the appropriate matches on its data graph and return the results on demand.

How can consumers of Linked Data request a particular format (e.g. JSON-LD)?

Software that wishes to retrieve Linked Data can use HTTP Content Negotiation to request the preferred format, by specifying a sequence of MIME types and associated preferences.

If the Linked Data is not available in the requested format, a number of tools exist, which can convert Linked Data from one format into another format, without loss of information or meaning.

Such tools include http://rdf-translator.appspot.com/ and software libraries in various programming languages, e.g. http://rdflib.net.

Consumers of Linked Data can also make use of SPARQL endpoints to request Linked Data that matches their SPARQL queries.

Why are we advocating the use of JSON-LD?

In this document, we recommend using JSON-LD because:

- Use of JSON-LD requires less detailed analysis / knowledge of the structure and layout of the human-readable web page, such as the nesting of elements within the page.
- The block of JSON-LD is largely decoupled from the rest of the web page and as such, it is possible to modify the visible layout of the page without needing to make changes to the JSON-LD block, so long as the structured data in the JSON-LD block still accurately corresponds to the human-readable information in the page.
- It is much easier and more scalable for GS1 work group to provide some worked examples of JSON-LD markup in a way that can easily be adapted by various user companies and their solution providers, rather than trying to develop individual customised examples using inline markup, for which users might have more difficulty in relating to the necessary modifications to their existing web pages.
- JSON-LD is considered to be less brittle than inline markup such as RDFa or Microdata. For example, using RDFa or Microdata, an image of a product appearing within the web page might be annotated with a property such as `http://schema.org/image` using the following markup example.

```
<div about="http://example.com/id/gtin/05011476100885" typeof="http://schema.org/Product">
  <img src="images/productXYZ.jpg"  property="http://schema.org/image">
</div>
```

However, if someone then wraps a hyperlink around an image and changes the markup to be as shown below, the interpetation of the RDFa markup changes.

```
<div about="http://example.com/id/gtin/05011476100885" typeof="http://schema.org/Product">
  <a href="promotion.html"><img src="images/productXYZ.jpg"
property="http://schema.org/image"></a>
</div>
```

In the example above, the image than becomes an image for the hyperlinked promotion page, instead of an image of the product, because the `href` value of the hyperlink overrides the subject specified in the `about` attribute of the parent `<div>` container.

JSON-LD is not susceptible to this brittleness caused by the addition of hyperlinks.

Which frameworks are available for serving Linked Data?

Linked Data can be served using an existing web server or it can be served using a dedicated Linked Data framework. These include commercial implementations as well as free or open source implementations, including those based on the W3C Linked Data Platform (LDP). Further information about implementations is available at http://www.w3.org/wiki/LDP_Implementations

What is Product Master Data?

Product master data typically consists of the specifications or attributes of a product that are stable over time and apply to all instances of that product class, i.e. every individual product package having that same GTIN barcode number can be expected to share the same characteristics that are described through master data. Master data can include information about material composition or ingredients, nutritional information, power consumption and technical specifications, as well as information about accreditations (e.g. environmental, ethical or dietary claims), as well as information about allergens that might be contained in the product - or other consumer safety information.

Typically master data is created by the brand owner or manufacturer and shared with the retailer of the product, so that consumers have access to this information even when they buy products online. For some kinds of product master data involving accreditations, independent accreditation

agencies might also play a role in contributing their assertions about the product, which can be embedded or referenced from the product master data.

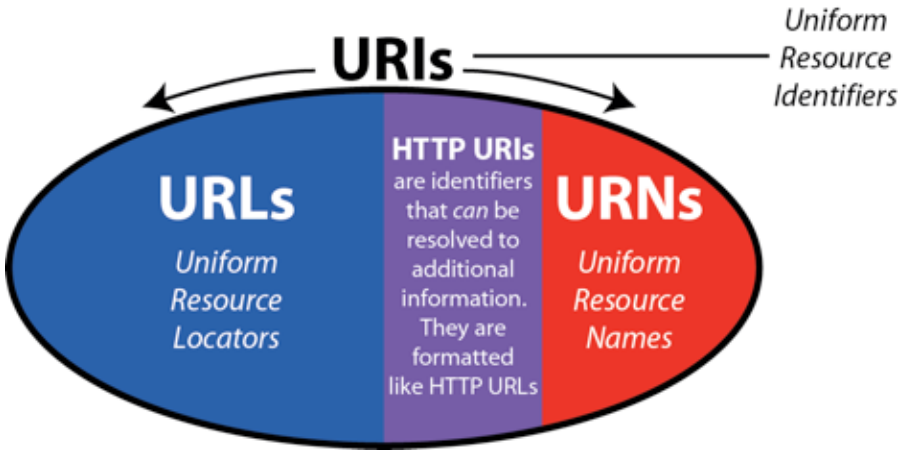
In the GS1 SmartSearch group, our initial focus is on the use of Linked Data technology to enable brand owners, manufacturers and retailers to publish product master data openly on the web, so that it is available for use by search engines, smartphone apps. We expect that the result of this will be to enable enhanced search listings for products and services, enable new business-to-consumer and consumer-to-business interactions, help consumers find the products they are really looking for, and help business understand how consumers are searching for products and the selection criteria that matter to them in their decisions.

Is the semantic web reliant on the Search Engine Optimisation (SEO) of websites?

No. Linked Data can be served within websites, using either inline markup (e.g. Microdata or RDFa) or a single block of structured data (e.g. JSON-LD) but it is also possible to serve the structured data directly using web server technology or other dedicated Linked Data mechanisms (such as RDF triple stores with SPARQL endpoints), so that search engines, smartphone apps and other software can make a request just for the structured data, without requesting the web page. There are some existing conventions and best practices for how to do this.

Having said that, some websites still make use of JavaScript or Flash for the navigation within the website. This can mean that search engines and other software is unable to interpret the JavaScript or Flash and therefore unable to discover all of the pages in the website that might contain structured data. It is therefore recommended to check whether someone can still navigate through the website even when their web browser has JavaScript and Flash switched off. If this is not possible, then it may be better to re-think how the navigation is done and to use more modern approaches to navigation toolbars and sidebars using HTML5 and CSS, rather than relying upon using JavaScript to do image rollovers and highlighting. If the HTML source code for your website includes many JavaScript 'onClick' handlers, then it might be a good idea to think about replacing these with regular hyperlinks (e.g. `<a href="….">`) since these will be more accessible to search engine crawlers and other software, without needing to understand any of the customised JavaScript code that was written for the individual website.

What is the difference between a URI and a URL?



URIs are Uniform Resource Identifiers.

URLs are Uniform Resource Locators (e.g. web addresses) and are used for retrieving information.

URNs are Uniform Resource Names (e.g. EPCs are canonically expressed as Pure Identity URIs using URN notation) and are used for globally uniquely naming things - but they have no obvious mechanism for retrieving information.

All URLs are URIs. All URNs are URIs. URIs are the 'union' of URLs and URNs.

Why do we use URIs?

Linked Data / Semantic Web technology makes extensive use of HTTP URIs, which can function either as names (like URNs) or as locators (like URLs).

You cannot always tell from looking at an HTTP URI whether it is serving the role of a name or a locator. However, you can make an HTTP GET request for that HTTP URI.

If it is serving a role as a name for a real-world thing or place, then that real-world thing or place cannot be delivered to you via the web, so the web server that handles that HTTP URI does the next best thing and returns an HTTP 303 'See Other' response code, together with a corresponding HTTP URI that works as a locator. Your web browser will then make a second request for that locator HTTP URI and obtain an information representation (e.g. web page) of the real-world thing or place.

Linked Data usually consists of RDF triples consisting of a Subject, Predicate and an Object. The HTTP URIs that work like names can be used in the Subject, Predicate or Object positions of RDF triples. The HTTP URIs that work like locators are used to retrieve a collection of RDF triples. So for example,

- `http://dbpedia.org/resource/Brussels`

  is an HTTP URI that serves as a name and is used in the subject of many RDF triples at DBpedia for facts about Brussels, capital of Belgium.

- `http://dbpedia.org/page/Brussels`

  is an HTTP URI that serves as a locator and is used to retrieve a web page containing a collection of those RDF triples from DBpedia.

When you type `http://dbpedia.org/resource/Brussels` into your web browser, DBpedia returns an HTTP 303 'See Other' response code and suggests that your web browser request `http://dbpedia.org/page/Brussels` instead. By doing so, DBpedia is saying "Sorry - I can't deliver Brussels to you via the web - try requesting this page of information about Brussels instead."

Does the URI replace GTIN?

No. We expect that for many years, the Global Trade Item Number (GTIN) will provide the primary key for identifying products at Point of Sale systems or accessing information via GDSN. However, the GTIN is simply a numeric string. Unlike a URI, it does not indicate any obvious or trivial mechanism for retrieving data about the object identified by a GTIN. In contrast, an HTTP URI looks like a web address or URL and can be configured to behave like one, returning data in response to a web request. HTTP URIs complement GTINs.

How do we develop a structured data mark-up template?

From our experience, it will be best to do this using a single block of JSON-LD either within the <head> block of the HTML page or as the last child element within the <body> block of the HTML page, instead of using inline markup such as RDFa or Microdata.

The reason is that it is very easy for RDFa markup to become broken. For example, an HTML image tag <img> might contain an RDFa attribute such as

`property="schema:image"`

and this is understood as meaning that this image is being said to be representative of something in an enclosing block of HTML that was identified using an RDFa attribute such as

`about="http://nestle.com/id/05011476100885"`

That all works fine until somebody else puts a hyperlink around the image, perhaps to link to a promotion or even open a pop-up window with additional views of the product.

When they do that, if they are not very careful to have asserted an explicit attribute of `about="http://nestle.com/id/05011476100885"` within the <img> tag, the image will be considered to be a `schema:image` of the new hyperlink, rather than a `schema:image` of the product.

It's also much more tricky to do RDFa correctly in the first place because the structure of the web page and its various nested blocks needs to be carefully considered.

It is much easier for GS1 to provide a JSON-LD template (perhaps one for food products, one for textiles, etc.) that can then simply be populated with the actual values (e.g. weight, colour, size, nutritional info, etc.) without either GS1 nor the retailer or brand owner being concerned about where that information appears within the web page.

However, it is important that the information that appears in the JSON-LD block is consistent with the information appearing in the web page.

At present, Google are still in the process of fully recognising JSON-LD as acceptable markup, but we have had discussions with them to explain why JSON-LD will be much more practical to deploy than inline markup such as RDFa or Microdata. We are also encouraging them to improve their support for JSON-LD in their Google Structured Data Testing Tool at http://www.google.com/webmasters/tools/richsnippets although other tools are available for testing, including http://linter.structured-data.org/

What is JSON?

JSON is an abbreviation for JavaScript Object Notation. JSON is compact way of exchanging structured data objects (lists, key-value pairs). JSON is already used in websites and smartphone apps for exchanging fragments of data between the web browser or app and the backend server (e.g. for auto-suggest, auto-complete etc.) without reloading the page.

What is JSON-LD?

JSON-LD is an abbreviation for JavaScript Object Notation for ***Linked Data***. JSON-LD provides a way to make pieces of JSON interoperable with each other, by mapping locally-defined keys (properties) to global URIs. JSON-LD also provides a way to include structured data in a web page as a single block. JSON-LD is less fiddly and less brittle than inline markup such as RDFa or Microdata. Unlike the latter, JSON-LD appears within a `<script>` element within the <head> or <body> element of a web page, but does not require inline annotations interspersed with the visible content.

What are JSON-LD Templates?

A JSON-LD Template is a single block of machine-interpretable structured data that can be placed within the <head> or <body> element of a web page or served as standalone structured data.

Where can I learn more about JSON-LD?

For more information about JSON-LD, please see:

- http://json-ld.org/ JSON-LD site & playground
- http://www.w3.org/TR/json-ld/ W3C standard
- http://youtu.be/vioCbTo3C-4 video intro
- http://www.slideshare.net/gkellogg1/json-for-linked-data

What do we mean by a 'trusted source of data'?

Trusted source of data refers to techniques that provide an assurance that the data was provided by an organisation that had the authority to provide that data, such as the brand owner or manufacturer of a product. This is in contrast with data from untrusted or non-authoritative sources, such as crowd-sourced data about products. Because most brand owners who apply barcodes to products lease a GS1 Company Prefix from a national GS1 member organisation (such as GS1 UK), GS1 is in a unique position to know which brand owner is associated with a given product barcode (GTIN – Global Trade Item Number) – and to be able to confirm whether data about a product came from a trusted source, typically the brand owner.

What role does standardisation have to play in the semantic web?

Standardisation plays a critical role in promoting interoperability and reducing ambiguities and incompatibilities in the web of data. The World Wide Web Consortium (W3C) has overseen the development of many of the fundamental technical standards that provide the framework for exchanging structured data using semantic technologies. Most of these standards are supported by commercial and free or open source implementations. For web vocabularies, there are some

standardised ontologies such as Dublin Core [http://dublincore.org - see also IETF RFC 5013, http://tools.ietf.org/html/rfc5013 , ISO 15836:2009], as well as web vocabularies (such as schema.org) that were initially developed outside of a standards process, but which are now being further developed and extended within a collaborative community process, with involvement from the W3C. [http://www.w3.org/wiki/WebSchemas]. For over 40 years, GS1 has brought together a community of brand owners, manufacturers, distributors and retailers in a number of industry sectors to work together on common standards for exchanging information within supply chains. The GS1 community has already developed extensive detailed data models and data dictionaries for describing products, services and organisations and the GS1 Digital initiative and GS1 SmartSearch work group is now making these available as a web vocabulary for use with Linked Data technologies.