Customer Experience Digital Data Acquisition 0.5

Revision Date: May 17, 2013

DRAFT Community Group Final Specification

Latest version:

http://www.w3.org/TR/PLACEHOLDER

Authors:

Individual	**Organization**
Miles Fender Murray Williams	**Accenture**
Vinay Goel Reza Jalili Dave McNamee	**Adobe**
Jay Myers	**Best Buy**
Jon Revill	**Blue Cross and Blue Shield of North Carolina**
Blane Sims Jared Vestal Eric Lunt	**BrightTag**
Patrick Wyatt	**Criteo**
Anna Long Eric Feinberg	**Digital Analytics Association**
Peter Loveday	**Digital Window**
Daniel Karpantschof	**Economist Digital, The**

Mark Prince Josh Goodwin	**Ensighten**
Mathieu Jondet	**Eulerian Technologies**
Laura Holmes Brian Kuhn Justin Cutroni Vishal Goenka	**Google**
Brian Hendrixson	**HSN**
Viswanath Srikanth Eliot Towb Hutch White	**IBM**
Lee Isensee	**Localytics**
Gagan Kanwar Emilie Laffray Nicholas Gadacz	**Marin Software**
Anton Gething	**nToklo**
Harry Hurst Ian McCaig Stephen Elliott	**QuBit**
Keith Watkins	**Red Hat**
Jonathan Conway	**Reevoo**
Oliver Schiffs	**SapientNitro**
Alexander Dean	**Snowplow Analytics**
Toby Doig	**TagMan**
Ali Behnam	**Tealium**

Abstract

This specification describes a method for surfacing Customer Experience Digital Data on a Web/Digital resource as a set of JavaScript Objects, and also specifies the parameters for communicating this data to digital analytic and reporting servers.

Status of this Document

This is a DRAFT of the Final Specification Document of the W3C Customer Experience Digital Data Community Group.

Table of Contents

1. Introduction

Collection and analysis of visitor behavioral and demographic data has become an integral part of web design and website success. This data is central to site performance analysis, dynamically tailoring site content to visitor activity and interest and retargeting visitors based on behaviors.

Increasingly, multiple vendors are involved in the data collection process on a given digital property, and each has a solution that needs to be implemented on the web page. As a result, page design has become more complex and development cycles lengthen as different requirements for data surfacing and formatting are added to the design process. Further, changing or adding vendors typically requires that the development team change page design to accommodate vendor-specific requirements. Common data items must be continually surfaced in different ways - and each design requirement is a custom effort, further lengthening development cycles. Companies are searching for a simpler, flexible and standard method to surface this common data on the web and across their digital properties.

This document details the specification for a standard data layer that collects this valuable user interaction information for subsequent use in analysis and reporting. The information in this document will be relevant to Web Analysts, Website Implementation Engineers as well as Marketing Professionals who need to understand user experience data that is being gathered.

2. Rationale and Design Goals

Customer Experience digital data items that are tracked and captured by different vendors are commonly understood elements used in digital analytics, but vendor specific format requirements and code assignments create design complexity and vendor dependency in site design.

As a simple instance, vendorA may capture some digital data for page details as a concatenation of ‘PageID + PAGENAME + PAGECATEGORY’ while vendor B may capture the same through distinct variables vendorB.page.pageId, vendorB.page.pageName, vendorB.page.pageCategory. Further, the names of the variables, the data structure name & types and methods of extensibility all vary on a per-vendor basis. Frequently, custom code must be written to capture data to meet a given vendor’s requirements.

Cumulatively, differences of this type between vendors permeate across all relevant customer experience digital data objects including Order, Shopping Cart, Registrant and more, increasing the complexity of customer experience digital data management at the site.

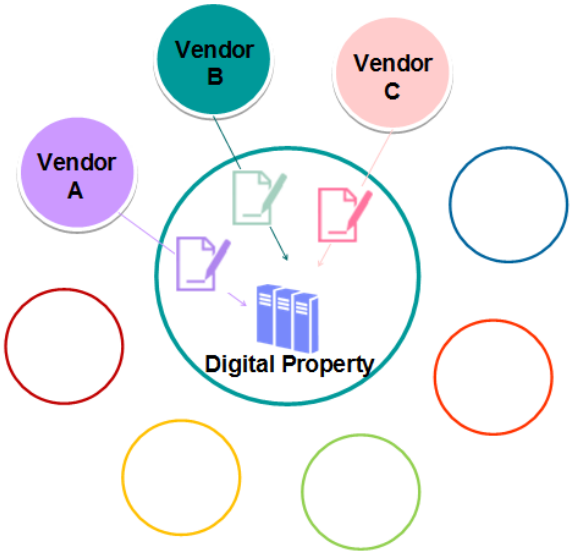

Figure 1: Vendor specific tags on a digital property

To this end, a Standard data object that represents the common data elements in a standard way will allow development teams to implement standard design structures that populate the Standard data object. Vendor code placed on the page would reference that standard structure, simplifying the process of on-boarding or changing vendors, and shaving off expensive development cycles.

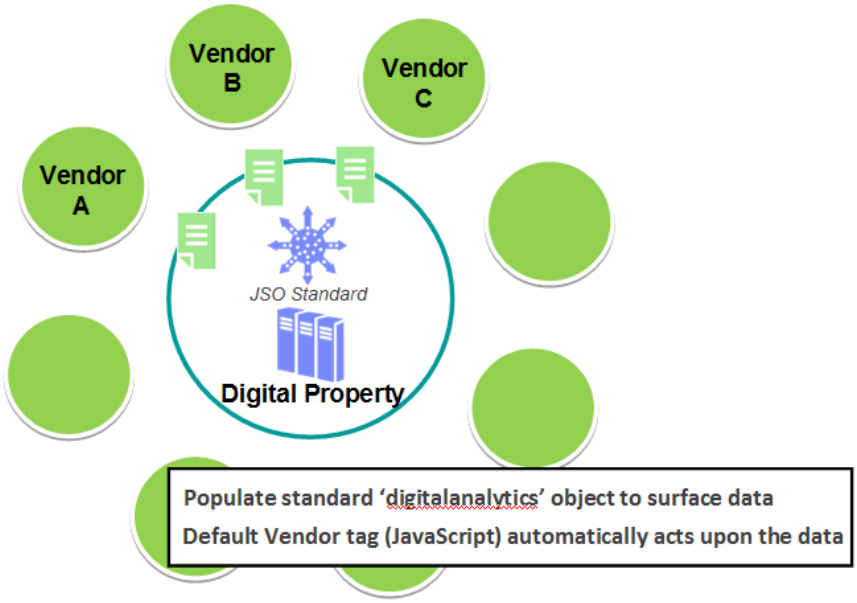

Figure 2: Adherence to standard data object accelerates deployment, simplifies site management

The proposed Standard data object is a JavaScript object because of ubiquitous support for JavaScript in web browsers and web based applications, as well as in other forms of digital properties like mobile & kiosks and so forth.

The use of a JavaScript object means that the data is not embedded in the html markup and would not affect page registration or performance. Developers would only need to populate the data fields in the object that are applicable to the page, keeping the size and complexity of the object to a minimum. That coding would never change regardless of page requirements vis-à-vis vendor additions or deletions unless new data elements were required. Because the data is a standard object, vendors who recognize the object would provide code that references that object, requiring no modification of the page other than to drop the vendor code container onto the page.

This standard will thus yield a shorter development cycle using fewer resources and there will be a savings to the client in time and money. The addition or deletion of vendors will be independent of considerations of page redesign, allowing decisions to be made based on vendor efficacy rather than cost of migration.

3. How to Read this Document

To understand the value of the specification, it is useful to go through Sections 1, 2, 3 and 4 to get a high level understanding behind the rationale, applicability and use of the specification. Section 5 talks about how the code will appear on your digital property, and Section 6 discusses Privacy implications. Sections 7 & 8 take a deep dive into the specific objects and their attributes. Section 9 & 10 wrap up with additional examples around the specification.

Also note that **none of the objects in the specification are designed to be required** - use of each object is at the discretion of the digital property owner. However, once an object has been selected for use, the object must adhere to the structure as specified in this document to remain conformant.

The term **reserved** is encountered in the specification when defining certain Object Literals. In this context, names of certain values are reserved for specific use within the Object Literal, and these values should be populated for intended purpose only to maintain the integrity of the specification.

4. The Specification through Examples

This section will introduce the specification through detailed examples.

5. Instrumentation/Deployment of the Specification

This section will discuss deployment of the JSO on the page.

Two data points to anchor this section:

- Instantiation of the data layer should be done before any use of the data layer (ideally as high on the page as possible), e.g. if you plan to use an analytics tag, you should place the data layer above the analytics tag; if you plan to use a TMS, you should place the data above the TMS snippet

- Recommend placing the data layer directly in the page, as opposed to an included script; this will allow anyone trying to consult on data collection implementation to access the data layer easily when inspecting the source code of your page

6. Privacy Implications

This section will detail recommendations to help with privacy related access issues for siteowners.

7. The Customer Experience Digital Data Object

7.1 The Root JavaScript Object

This section is normative

The root JavaScript Object (JSO) MUST be digitalData, and all data elements used in digital analytics MUST derive off of this JavaScript Object either directly or indirectly.

The following sub-objects are defined directly off of the digitalData object. and the following subsections detail each of these subobjects.

ObjectName
digitalData.pageIDentifier

digitalData.page
digitalData.product[n]
digitalData.cart
digitalData.transaction
digitalData.event[n]
digitalData.component[n]
digitalData.user[n]
digitalData.version

7.2 Page Identifier Object

This section is normative

Page Identifier is among the most widely used web analytic data element, and is among the top level web analytic objects. A Page Identifier MUST have the following Object Name & Type.

ObjectName	**Type**	**Comment**
digitalData.pageIDentifier	String	ID content area specific to environment; to deal with dev, staging, and production - unique among environments

7.3 Page Object

This section is normative

Page Object carries significant details about the page, and the most commonly used data elements are captured by the specification below. Page MUST have the following Object Name & Type.

ObjectName	**Type**	**Comment**
digitalData.page.pageID	String	Page ID
digitalData.page.pageName	String	
digitalData.page.pageCategory	Object Literal = { primaryCategory:'FAQ	Because of the wide range of methods for categorization, an object literal works best here.

	Pages' subCategory1: 'ProductInfo', PageType: 'FAQ' }; **Reserved**: primaryCategory: String	The first name, primaryCategory is reserved. All other values are optional and should fit the individual implementation needs in both naming and values passed.
digitalData.page.searchTerm	String	Internal Search Term
digitalData.page.searchResult	Integer	Number of Internal Search Results
digitalData.page.destinationURL	String	Destination URL
digitalData.page.referringURL	String	Referring URL

digitalData.page.attributes	Object Literal
	={
	SysEnv: 'mobile',
	Variant: '2',
	Version: '1.14'
	Breadcrumbs:
	'home,Products,haircare,
	};
	
	Reserved:
	SysEnv: String
	Variant: String
	Version:String
	Breadcrumbs:String
		The object provides extensibility to the Page object. There are four reserved names, SysEnv,Variant,Version, and Breadcrumbs. All other values are optional and should fit the individual implementation needs in both naming and values passed.

7.4 Product Object

This section is normative

Product Object carries details about a particular product with frequently used attributes listed below. Product Object MUST have the following Object Name & Type.

ObjectName	**Type**	**Comment**
digitalData.product.productID	Object Literal = { prodID: 'rog300', productName: 'Rogaine', description: 'Hair Regrowth'	The object provides extensibility to the Product object. There are four reserved names: prodID,

	productURL: 'http://site.com/r.html' manufacturer:'Pharma' size:'300ml' }; **Reserved:** prodID:String productName:String description:String productURL:String	productName, description, and productURL. Other names optional and should fit the individual implementation needs in both Naming and values passed.

digitalData.product.productCategory	Object Literal
	= {
	primaryCategory: 'Haircare',
	subCategory1: 'Men's',
	ProductType: 'Thining Hair Treatments'
	}
	
	Reserved:
	primaryCategory: String
		Because of the wide range of methods for categorization an object literal works best here. The name primaryCategory is reserved . Other names are optional and should fit the individual implementation needs in both naming and values passed.

digitalData.product.linkedProducts	Object Literal ={ productID1: value from digitalData.product.productID.prodID; productID2: value from digitalData.product.productID.prodID .... }	Object Literal value containing lists of Linked Product IDs.
digitalData.product.attributes	Object Literal	Any additional dimensions related to the product as a name-value pair in an Object Literal

7.5 Cart Object

This section is normative

ObjectName	**Type**	**Comment**
digitalData.cart.cartID	String	ID associated with a shopping Cart

digitalData.cart.price	Object Literal
	= {
	basePrice:2.00,
	vouchercode:'Alpha',
	voucherdiscount:.50,
	currency: 'EUR',
	tax: '2.00',
	shipping: '1.14'
	shippingmethod: 'UPS'
	priceWithTax:
	cartTotal:
	};
	
	Reserved:
	basePrice: Number
	voucherCode:String
	voucherDiscount:
	Number
	currency: String
	tax: Number
	shipping: Number
	shippingmethod: String
	priceWithTax: Number
	cartTotal:
		The object provides extensibility to the Cart Price object. The name basePrice is reserved. All other names are optional and should fit the individual implementation needs in both naming and values passed. This object is duplicated in the Items object as some vendors track this information on each item.

digitalData.cart.items[n]	Array	List of Items in the Cart

digitalData.cart.items[n].
productID
	Object Literal
	= {
	prodID: 'rog300',
	productName: 'Rogaine',
	description: 'Hair
	Regrowth'
	productURL:
	'http://site.com/r.html'
	manufacturer:'Pharma'
	sku:'RG003'
	size:'300ml'
	};
	
	Reserved:
	prodID:String
	productName:String
	description:String
	productURL:String
		Same as productID defined under Products object.
		
		The object provides extensibility to the Items object. The names prodID, productName, description, and productURL are reserved. All other names are optional and should fit the individual implementation needs in both Naming and values passed.

digitalData.cart.items[n].	Object Literal	Same as productCategory under Product Object.

productCategory	= {
	primaryCategory: 'Haircare',
	subCategory1: 'Men's',
	ProductType: 'Thining Hair Treatments'
	}
	
	Reserved:
	primaryCategory: String
		Because of the wide range of methods for categorization an object literal works best here. The name primaryCategory is reserved. All other names are optional and should fit the individual implementation needs in both naming and values passed.

digitalData.cart.items[n].quantity	Number	QuantiyNumber of this particular item
digitalData.cart.items[n].price	Object Literal = { basePrice:2.00, vouchercode:'Alpha', voucherdiscount:.50, currency: 'EUR', tax: '2.00', shipping: '1.14' shippingmethod: 'UPS' priceWithTax: cartTotal: };"	Same as Price at the Cart Level. The object provides extensibility to the Cart item Price object. There may be duplication with the main Cart object but some vendors track this data on each individual item in the Cart. Other than the Reserved values, all others are optional and should fit the individual implementation needs in both naming and values passed.
digitalData.cart.items[n].linkedProducts	Object Literal	Same as under the Product Object. Object Literal value containing list of Linked Product IDs.
digitalData.cart.items[n].attributes	Object Literal	Same as under the Product Object. Object Literal value for additional dimensions of the item.

7.6 Transaction Object

This section is normative

ObjectName	**Type**	**Comment**
digitalData.transaction.transactionID	String	
digitalData.transaction.profileID	String	
digitalData.transaction.profileID.	Object Literal	The object

address	= {
	line1:'673 Mystreet',
	line2:'Apt 1',
	city:'Austin',
	stateProvince: 'TX',
	postalCode:'78610'
	country: 'USA',
	};
	
	Reserved:
	line1:String
	line2: String
	city:String
	stateProvince:String
	postalCode:String
	country:String
		provides extensibility to the purchaser object. There are six reserved names: line1, line2, city, stateProvince, postalCode, and country. Optional values (those not in the reserved list) should fit the individual implementation needs in both naming and values passed.

digitalData.transaction.profileID.
shippingAddress
	Object Literal
	= {
	line1:'673 Mystreet',
	line2:'Apt 1',
	city:'Austin',
	stateProvince: 'TX',
	postalCode:'78610'
	country: 'USA',
	shippingMethod:'UPS'
	};
	
	Reserved:
	line1:String
	line2: String
	city:String stateProvince:String
	postalCode:String country:String
	shippingMethod:String
		The object provides extensibility to the purchaser object. There are seven reserved names: line1, line2, city, stateProvinde, postalCode, country, and shippingMethod. Optional names (those not in the reserved list) should fit the individual implementation needs in both naming and values passed.

digitalData.transaction.total	Object Literal ={ totalPrice:2.00, voucherCode:'Alpha',	The object provides extensibility to the Transaction

	voucherDiscount:.50, currency: 'EUR', paymentType:'VISA' tax: '2.00', shipping: '1.14' shippingMethod:'UPS' }; **Reserved:** totalPrice:Number voucherCode:String voucherDiscount:String currenty:String paymentType:String tax:Number shipping:Number shippingMethod:String	object. There are eight reserved names: totalPrice, voucherCode, voucherDiscount, currency, paymentType, tax, shipping, shippingMethod. Optional names (those not in the reserved list) should fit the individual implementation needs in both naming and values passed.
digitalData.transaction.attributes	Object Literal	Provides extensibility
digitalData.transaction.items[n]	Array	List of items in the transaction

digitalData.transaction.items[n].
productID
	Object Literal
	={
	prodID: 'rog300',
	productName: 'Rogaine',
	description: 'Hair Regrowth Treatment'
	productURL:
	'http://somesite.com/Products/rogain300.html'
	manufacturer:'Pharmecutical Company'
	sku:'RG003'
	size:'300ml'
	};
	
	Reserved:
	prodID:String
	productName:String
	description:String
	productURL:String
	manufacturer:String
	sku:String
	color:String
		The object provides extensibility to the Transaction object. The name prodID, is recommended and reserved. Other reserved names are productName, description, productURL, manufacturer, sku, color and size. All other values are optional and should fit the individual

	size:String	implementation needs in both Naming and values passed.

digitalData.transaction.items[n].
categoryID
	Object Literal
	={
	primaryCategory: 'Haircare',
	subCategory1: 'Men's',
	productType: 'Thining Hair Treatments'
	};
	
	Reserved:
	primaryCategory:String
		Because of the wide range of methods for categorization and object literal works best here. The name primaryCategory is recommended and reserved, all other names are optional and should fit the individual implementation needs in both naming and values passed. W3C should have a list of best practices but not require a particular structure.

digitalData.transaction.items[n]. quantity	Number	

digitalData.transaction.items[n].
price
	Object Literal
	={
	basePrice:2.00,
	voucherCode:'Alpha',
	voucherDiscount:.50,
	currency: 'EUR',
	tax: '2.00',
	shipping: '1.14'
	shippingMethod:'UPS'
	};
	
	Reserved:
	basePrice:Number
		The object provides extensibility to the Transaction Items object. There may be duplication with the main Transaction object but some vendors track this data on each individual item in

	voucherCode:String voucherDiscount:Number currency:String tax:Number shipping:Number shippingMethod:String	the cart. There are seven reserved names, basePrice, voucherCode, voucherDiscount, currency, tax, shipping, shippingMethod. All other names are optional and should fit the individual implementation needs in both naming and values passed. W3C should have a list of best practices but not require a particular structure.
digitalData.transaction.items[n]. linkedProducts	Object Literal	Contains lists of linked Product ID's
digitalData.transaction.items[n]. attributes	Object Literal	Provides extensibility

7.7 Event Object

This section is normative

ObjectName	**Type**	**Comment**
event[n]	Array	
event[n].eventID	Object Literal ={ eventName:'Add News Portal', eventAction:'addportal', eventPoints:200,	The object provides extensibility to the event object. There are seven reserved names, eventID, eventName, eventAction, eventPoints, type, timeStamp and effect. The name eventID is required. All other names are optional and should fit the individual implementation needs in both

	type:'contentModifier', timeStamp:nnnnnnnnn, effect:'include portal 1234' }; **Reserved:** eventName:String eventAction:String eventPoints:Number type:String timeStamp:number effect:String	Naming and values passed. W3C should have a list of best practices but not require a particular structure.

event[n].eventCategory	Object Literal
	={
	primaryCategory: 'Portal',
	subCategory1: 'dashboard',
	}
	
	Reserved:
	primaryCategory:String
		Because of the wide range of methods for categorization and object literal works best here. The names primaryCategory is recommended and reserved, all other names are optional and should fit the individual implementation needs in both naming and values passed. W3C should have a list of best practices but not require a particular structure.

event.attributes	Object Literal
	={
	cause: 'Portal Selection From Dropdown',
	effect: 'Add News Portal',
	};
	
	Reserved:
	cause:String
	effect:String
		Object literal to provide object extensibility and more granular tracking. There are two reserved names, cause and effect. W3C should have a list of best practices but not a required structure.

7.8 Component Object

This section is normative

ObjectName	**Type**	**Comment**
component[n]	Array	
component[n].componentID	Object Literal ={ compID: 'rog300v',	The object provides extensibility to the component object. The name compID is recommended and reserved. Other reserved

	componentName: 'How to use Rogaine', description: 'Hair Treatment Video' }; **Reserved:** compID:String	value Names are compID, componentName, and description. All other names are optional and should fit the individual implementation needs in both Naming and values passed. W3C should have a list of best practices but not require a particular structure.

component[n].category	Object Literal
	={
	primaryCategory: 'Haircare',
	subCategory1: 'VIDoes',
	productType: 'Flash Movie'
	};
	
	Reserved:
	primaryCategory:String
		Because of the wide range of methods for categorization and object literal works best here. The value, primaryCategory is recommended and reserved, all other names are optional and should fit the individual implementation needs in both naming and values passed. W3C should have a list of best practices but not require a particular structure.

component[n].attributes	Object Literal	Provides extensibility.

7.9 User Object

This section is normative

ObjectName	**Type**	**Comment**
user[n]	Array	Array of Users
user[n]. segment	Object Literal	
user[n].profile[n]	Array	Array of User Profiles.

user[n].profile[n].
profileID
	Object Literal
	={
	profileID: 'humanbeing12345',
	userName: 'me',
	};
	
	Reserved:
	profileID:String
	userName:String
		The object provides extensibility to the profile object. The value, profileID, is recommended and reserved. The other reserved name is userName. All other names are optional and should fit the individual implementation needs in both Naming and values passed.

user[n].profile[n].	Object Literal	There are six reserved names, line1, line2, city,

address	={
	line1:'673 Mystreet',
	line2:'Apt 1',
	city:'Austin',
	stateProvince: 'TX',
	postalCode:'78610'
	country: 'USA',
	};
	
	Reserved:
	line1:String
	line2:String
	city:String
	stateProvince:String
	postalCode:String
	country:String
		stateProvince, postalCode, and country. All other names are optional and should fit the individual implementation needs in both naming and values passed.

user[n].profile[n].social	Object Literal
	={
	twitter: 'soembody',
	twitterinfo: 'stuff',
	facebook: 'somebody1234'
	facebookinfo: 'morestuff'
	};
		The object provides extensibility to the profile object. All names are optional and should fit the individual implementation needs in both naming and values passed.

user[n].profile[n].attributes	Object Literal
	={
	userLogin: 'logmein',
	email:
	'somebody@somesite.com',
	language: 'chr'
	returningStatus: 'new'
	type:'premium'
	};
	
	Reserved:
	userLogin:String
	email:String
	language:String
	returningStatus:String
	type:String
		The object provides extensibility to the profile object. There are five reserved names: userLogin, email, language, returningStatus, and type. All other names are optional and should fit the individual implementation needs in both Naming and values passed.

7.10 Version Object

This section is normative

ObjectName	Type	Comment
version	String	

8. Extending the Specification

Normative section regarding extensions. Then followed by examples of extending the specification

9. Industry Specific Examples of Using the Specification

Industry specific examples – using the specifications, including its extension capabilities

10. Additional Examples of Using the Specification

This continues illustration of the specification through examples including Events, and other areas that we should illustrate.

At the end of the day, reading sections 3, 4, 7 & 8 should give the reader virtually everything they need to know about the specification without even going through the details of the specification.

11. Acknowledgements

Additional Contributors

12. References

RFC 2119: *Key words for use in RFCs to Indicate Requirement Levels* , S. Bradner, Author. Internet Engineering Task Force, March 1997. Available at http://www.ietf.org/rfc/rfc2119.txt.

RFC 4627 : *The application/json Media Type for JavaScript Object Notation (JSON), D. Crockford,* Author. Internet Engineering Task Force, July 2006. Available at http://www.ietf.org/rfc/rfc4627.txt.

13. Appendices

To be filled in as needed.