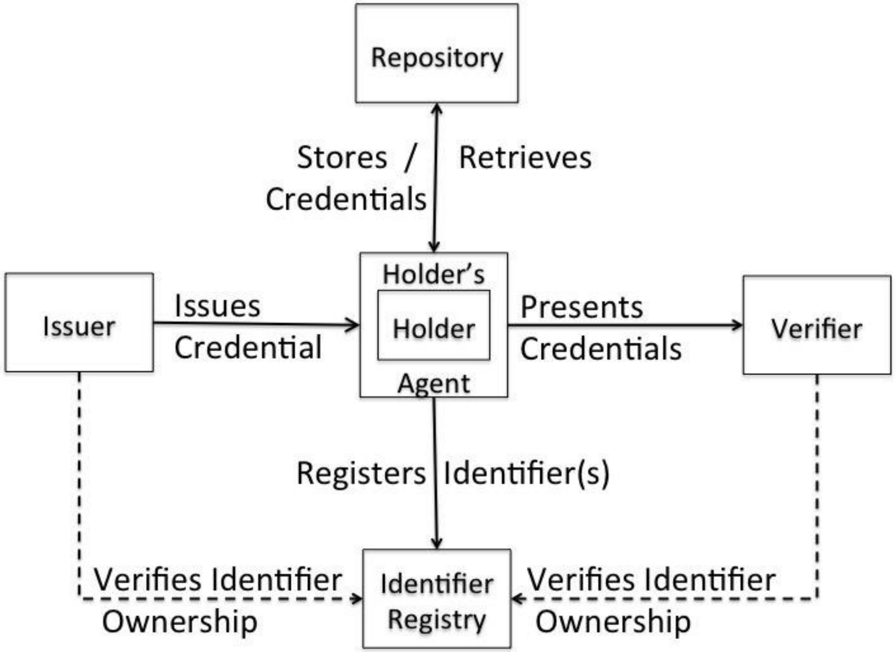
Primer for Dr. Ryan Wisnesky at the DIF Interop WG

Brent Shambaugh

Verifiable Credentials Lifecycle



https://w3c-ccg.github.io/vc-lifecycle/

Identifier Registry contains DIDs and possibly full or a hash representation of DID Documents and/or Schemas?

Decentralized Identifier Architecture

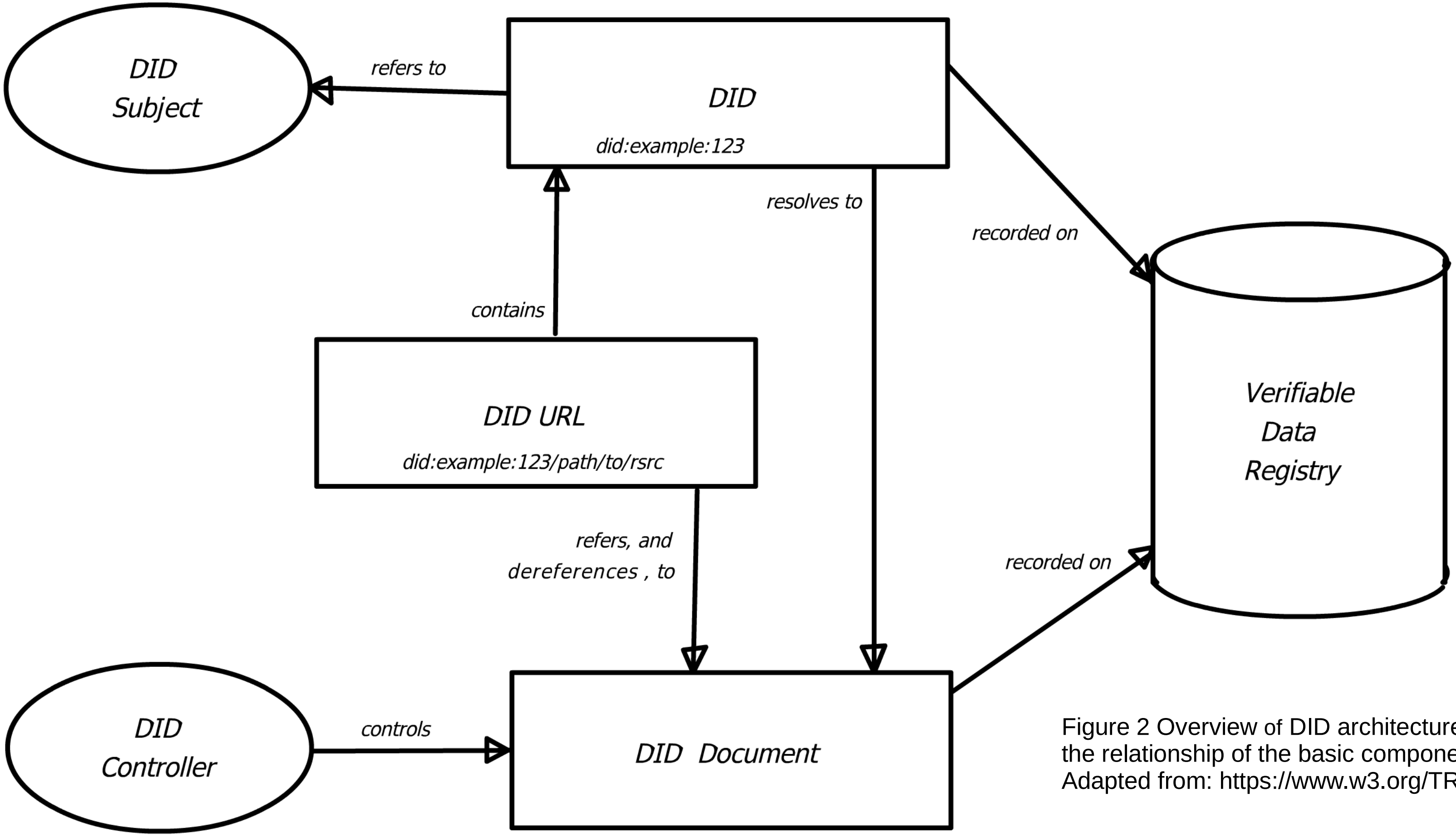
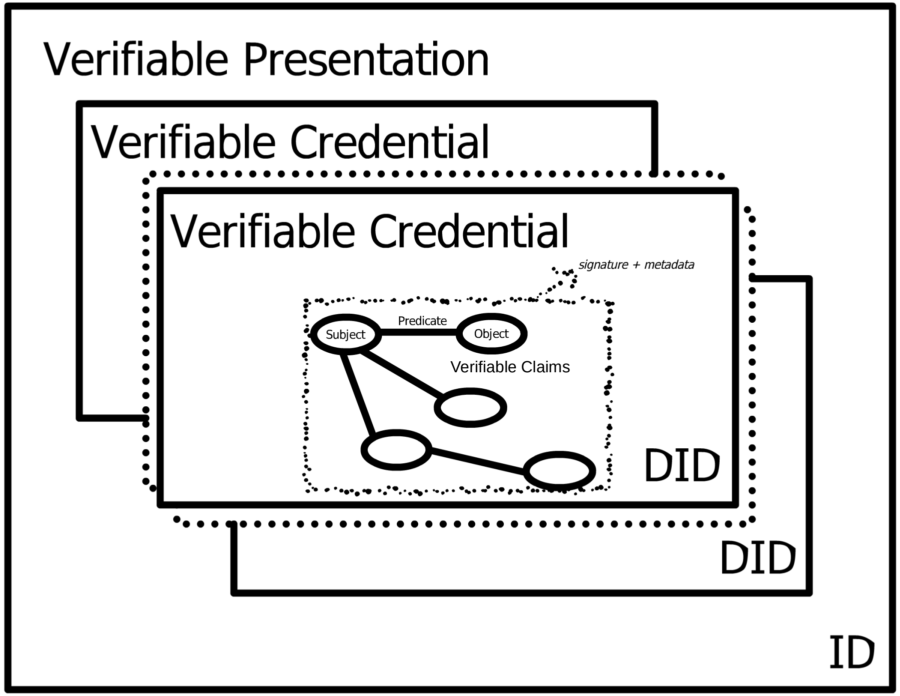
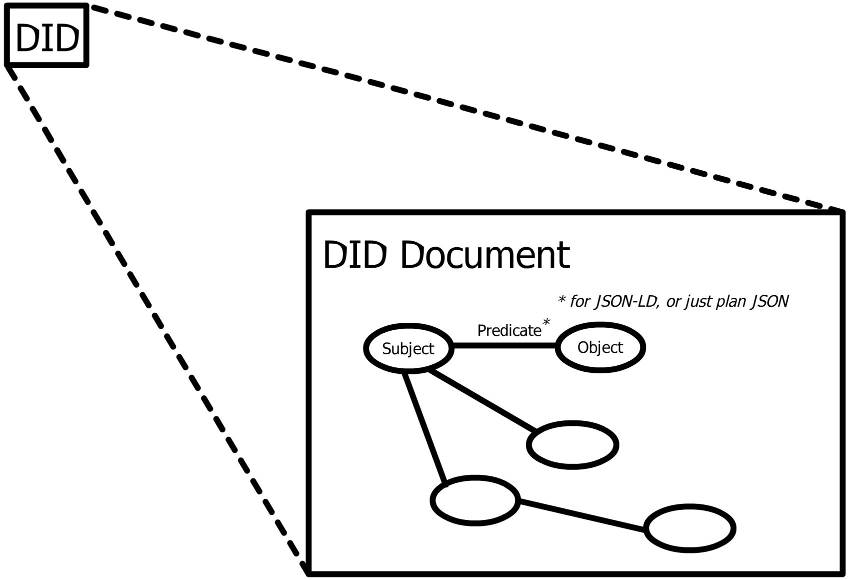


Figure 2 Overview of DID architecture and the relationship of the basic components. Adapted from: https://www.w3.org/TR/did-core/

Data Models





Inspired by:
https://www.w3.org/TR/did-core/

Inspired by:
https://www.w3.org/TR/vc-data-model/,
https://identity.foundation/presentation-exchange/spec/v2.0.0/

Proof Mechanisms

JSON Web Tokens [RFC7519] secured using JSON Web Signatures [RFC7515]
Data Integrity Proofs [DATA-INTEGRITY]
Camenisch-Lysyanskaya Zero-Knowledge Proofs [CL-SIGNATURES].
JSON Web Proofs - JWTs with Superpowers [JSON Web Proofs]

Signatures require Canonicalization
https://w3c-ccg.github.io/rdf-dataset-canonicalization/spec/index.html

[DATA-INTEGRITY] Verifiable Credential Data Integrity 1.0, M. Sporny et al.,
https://www.w3.org/TR/vc-data-integrity/
[RFC7519] The JavaScript Object Notation (JSON) Data Interchange Format, T. Bray, ed,
https://www.rfc-editor.org/rfc/rfc8259
[RFC7515] JSON Web Token (JWT), M. Jones et al.,
https://www.rfc-editor.org/rfc/rfc7519
[CL-SIGNATURES] A Signature Scheme with Efficient Protocols. Camenisch et al., IBM Research. Peer Reviewed Paper.
https://www.researchgate.net/publication/220922101_A_Signature_Scheme_with_Efficient_Protocols
[JSON Web Proofs] https://raw.githubusercontent.com/windley/IIW_homepage/gh-pages/assets/proceedings/IIW_33_Book_of_Proceedings.pdf,
https://github.com/json-web-proofs/json-web-proof

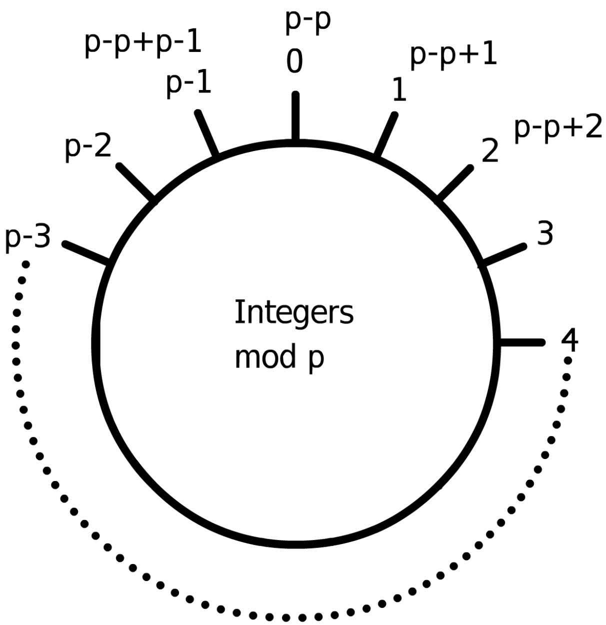
Cryptographic Signature

Algorithm	EdDSA	ECDSA	Yours
Generate	d_v private signing key generated from random integer (seed) $d_p = d_v * G$ public key $r = hash(hash(d_v + m)) \bmod q$ $R = r * G$ $h = hash(R + d_p + m) \bmod q$ $s = hash(r + h * d_v) \bmod q$	$P = k * G$ $r = P_x$ $s = k^{-1}(hash(m) + d_v * P_x) \bmod p$ k is a random secret number used once in the range [0...p-1] P_x is the x-coordinate of P p is the order of the subgroup of the points generated by G d_v is the private signing key m is the message G is the generator point Signature is not deterministic due the random number k	...
Validate	$h = hash(R + d_v + m) \bmod q$ $P_1 = s * G$ $P_2 = R + h * d_p$ $P_1 = P_2 ?$	$s_m = s^{-1} \bmod p$ is the modular inverse of s $R' = (hash(m) * s_m) * G + (r * s_m) * d_p$ if $R'_x = P_x$ the signature is valid d_p is the public key	...
Source	https://cryptobook.nakov.com/digital-signatures/eddsa-and-ed25519	Real World Cryptography, David Wong, Manning, pg. 143 - 144 https://cryptobook.nakov.com/digital-signatures/ecdsa-sign-verify-messages https://learn.saylor.org/mod/book/view.php?id=36341&chapterid=18920	

Groups in ECC

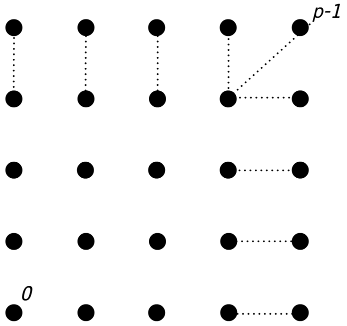
$y^2 = x^3 + ax + b (mod\ p)$

$y^2 = x^3 + ax^2 + x\ (mod\ p)$



Addition and subtraction are closed within the field

F_p p is a large number in a finite integer Field



The curves only have integers as points

The points on the curve can form a cyclic group

The total number of points on the curve is called the order, and this is a prime number.

Multiplication of an integer k by a generator G leads to another point on the curve.

$P = k * G$

If k is zero, then it is said to resolve to a point at infinity.

Curves can have one or more cyclic subgroups.

$n = h * r$

n order of the curve

h Curve co-factor

r Order of the subgroups

Talk about subgroups

Defintion of a Group

A group must have the properties:

Closure: For any **a** and **b**, ***a * b*** is also in the group

Associativity: For any **a**,**b**,**c** in a group, ***a * (b * c) = (a * b) * c***

Identity Element: For any **a** in the group ***a * 1 = a***

Inverse Element: For any **a** in the group, there is an $\mathbf{a^{-1}}$ as well, such that $\mathbf{a * a^{-1} = 1}$

Quoting, page 92, Real World Cryptography, David Wong, Manning Publications

Groups as Categories

“In particular, a group is a category with one object, in which every arrow is an iso. If G and H are groups, regarded as categories, then we can consider arbitrary functors between them $f : G \to H$. It is obvious that a functor between groups is exactly the same thing as a group homomorphism.”
pg. 72, chap 4, Category Theory, Steve Adowey

Definition of a Category

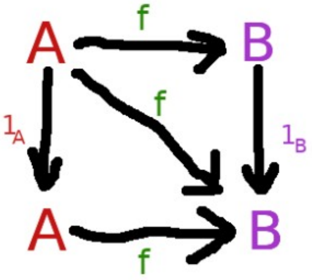
A category consists of:

- a collection of objects
- a collection of arrows

$$\underset{dom}{a} \xrightarrow{f} \underset{cod}{b}$$

Identity:

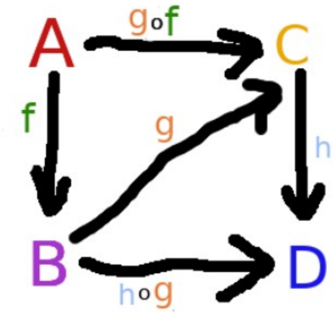
f o 1a = f = 1b o f



[2]

Associativity:

If morphism A → B is f, B → C is g , C → D is h then A → D is (h o g) o f = h o (g o f) = h o g o f

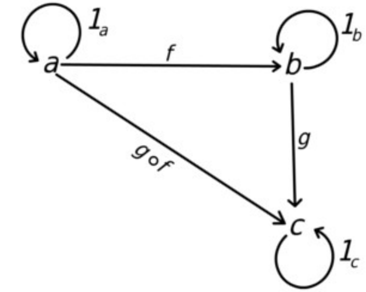


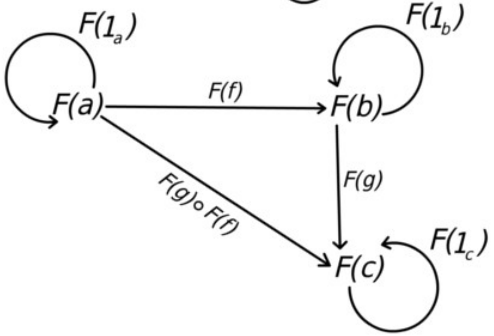
[2]

Uses of a Category

Definition of a Functor:

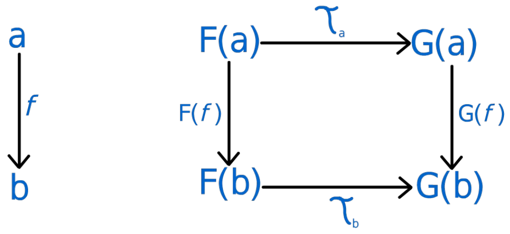
'A functor is a transformation from one category to another that "preserves" the categorical structure of its source'
pg. 194, The Categorical Analysis of Logic
-Goldblatt



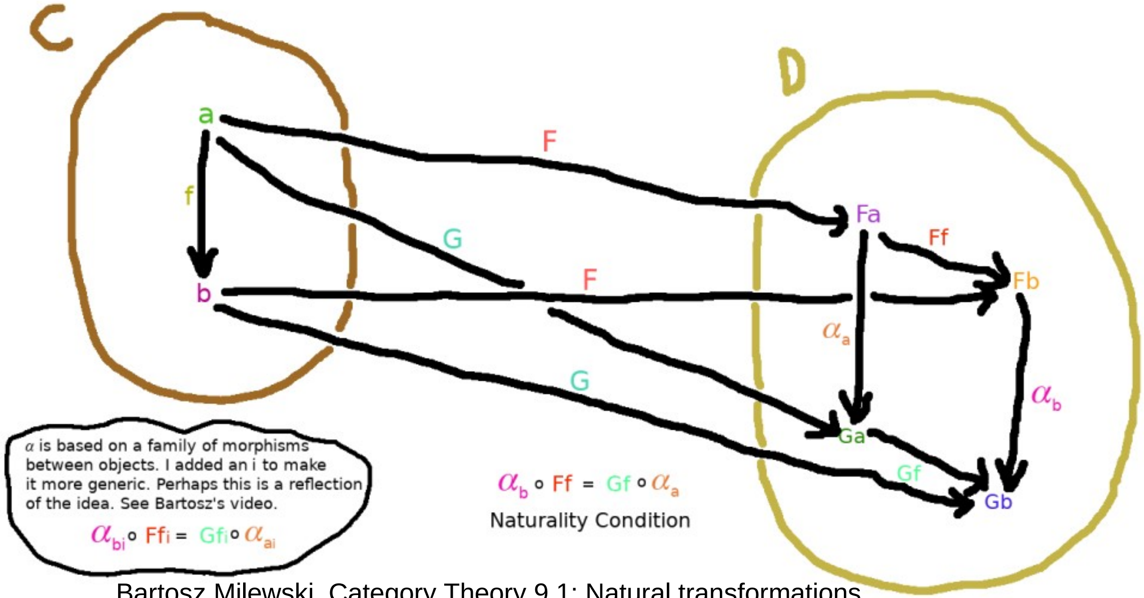


Natural Transformations Consider Functors to be Objects
pg. 198, Goldblatt

Definition of a Natural Transformation:



pg. 199, Goldblatt



Bartosz Milewski, Category Theory 9.1: Natural transformations
https://www.youtube.com/watch?v=2LJC-XD5Ffo

Resources to Consider

Syntactic Mapping : [Project Cambria]
Syntactic & Semantic Mapping: [LSA, Hydra]
Architecture: [OCA].
Category Theory w/ RDF & RDF Schema [Benjamin Braatz Thesis]

[Project Cambria] Project Cambria Overview with Geoffrey Litt and Peter van Hardenberg – Fission, https://fission.codes/blog/project-cambria-overview/

[LSA, Hydra] Layered Schema Architecture: https://github.com/cloudprivacylabs/lsa, Hydra, Transform Your Transformations: https://github.com/CategoricalData/hydra

[OCA] OCA Technical Specification | Overlays Capture Architecture, https://oca.colossi.network/specification/

[Benjamin Braatz Thesis] Formal Modelling and Application of Graph Transformations in the Resource Description Framework, Benjamin Braatz https://www.semanticscholar.org/paper/Formal-Modelling-and-Application-of-Graph-in-the-Braatz/b8c85a3e7a04020259ec9a58c7e5563033f52844