# IFC on Strange Matter

Strange Matter was born out of the needs expressed by the needs of both the buildingSMART community for future versions of IFC but also the large community around the build environment that is looking for a modern flexible way to connect data.

Strange Matter is an open protocol for working with distributed, heterogeneous data used before, during, and after all phases of design, construction and operations of the of the built environment.

Strange matter is format, vendor, and tool agnostic.

It is a way for people, processes, and tools with different requirements working together on design and construction projects to collaborate on data that has distributed ownership, comes from different sources, and that is continuously changing.

Strange Matter does this by providing a universal abstract concept of entity. That is the thing that people care about (whether it is a particular building, floor, facade, column, roadway, pile, rail bed, asset, or whatever) and for which more or less data may be available to different stakeholders over different periods of time, authored in different pieces of software.

Actual data is organized in components and relationships. Components are JSON headers that refer to data payloads, which can be in any format a user or tool generates. Relationships are defined in the same way as components, just without payloads, and can describe any kind of semantic relation between two components. A relationship between a component and entity is done by sharing a relationship with an Entity ID component.

# Design Criteria

Strange Matter was born from the key requirements of the industry. Chief among them is multiformat and multi standard support. It became clear that to reach this goal the protocol use to organize data had to be standard agnostic.

Beyond that the actual data representation was informed by broad community input most of which is captured here.

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Type** | **Design Solution** |
| Data is assembled across many sources. It changes over time, has many versions and states. A single thing might be simultaneously in design, construction and operations all at the same time. | Functional | Entity Component Model where the Components are Packets of Data. This is augmented by relationships that are also components. This allows data to be continuously added and related together while having no impact on the existing data |
| To describe the built world we need GIS, BIM, text, requirements, issues, inspections, lidar data, point cloud data, products, carbon, energy just to name a few. No format, standard etc. can ever describe all this data so we need to find a way that can connect standard, nonstandard and even proprietary data without taking ownership of its description |  | Strange matter is format, vendor, and tool agnostic. |
| Machine to Machine Readable |  | There have been attempts to package heterogeneous data in packages, but the data was only labeled or classified and didn’t allow machine to machine communication. To solve this Strange Matter has proposed robust methods of data description and self-description of payloads |
| The data we use to digitally describe the built world relies heavily on relationships. This is often to the service of the format more than it is to the users of the data. |  | Relationships are just as easy to create and compose as the data itself. This is a key requirement to incorporate automation, workflow and future machine learning capacities. |
| Data needs to work in many platforms, offline, online, archive, |  | Strange Matter is rather simple in design but meant to work as a complete protocol. Its well suited to be stored, access and created in many technologies. It’s simple enough that it can work as a file on disk, but it certainly can be use in SQL, NO-SQL, Graph and other formats.  The emergence of Columnar “File as Database” formats like [Apache Parquet](https://parquet.apache.org/) or [LanceDB](https://lancedb.github.io/lance/) that are become standard in the data science world are very intriguing as they mary very well the component based approach. This opens the door for native ML and automation capabilities directly on the data. |
| Data needs to work in many different governance models |  | Components are both small but also user scopable |
| Files first |  |  |
| Internet does last forever | Usability |  |
| Immutable |  |  |
| Reference not own… only add (application don't own the data) | |  |
| Share as You need (half) |  |  |
| Heiarchy is in the eye of the beholder | Security |  |
| definitoin is not instance data |  |  |
| Git |  |  |
| Data is independent of its container |  |  |
| Benefit for all Creators, Adders/Maintainers |  |  |
| Out of Order and Addititve |  |  |
| Protocol not an API… |  |  |
| Fine Control on Security/ Access |  |  |
| Strange Matter has one component (currently) |  |  |
| Classsifcation is for funtion/ Component Type is for content | |  |
|  |  |  |
| IFC on Strange Matter |  |  |
| location and positioning |  |  |
| Visual Display |  |  |
| Explicit Units |  |  |
| Bools not IFCBool |  |  |
| Simplify |  |  |
| Developer Experience |  |  |