The original implementation was organized by service type (event lookup, dataset overlaps, trigger statistics, ...)
- Each service gives access to all relevant data
- Following user requests, the User Interface has evolved into a system organized by data entity (events, datasets, collections, runs,...)
- Each entity gives access to all available services
- While the underlying implementation was still organized by service type
- New implementation (under development for Run 3) is organized by data entity
- Two prototypes were implemented, the Prototype 2 has been selected for the final implementation
- Navigation is provided via dynamic hierarchical graph-like overview of all available data and data collections
- Data are shown together with their relations, ownership, containment or overlaps
- Some actions are provided directly by the Event Index system, others are possible actions to perform (EI or external)
- In many cases, specialised views are offered for detailed data inspection
- In principle re-usable for any Graph-like data
- Customizable via stylesheets

Prototype 2

selected for Run 3 Event Index implementation

1. Create a prototype of the Element you want to search
2. Fill in known values
   a. You can use SQL for Phoenix part
   b. You may choose which backed (Phoenix, HBase or both) is used for searching and data filtering
3. Send to the ElementFactory
4. Get a set of satisfying Elements, with all values filled in from both Phoenix and HBase
5. Add Tags, Relations of Extensions to Elements
   a. Elements is a Relation
   b. Title is a Extension
6. Update via ElementFactory
   a. HBase will be updated

Prototype 1

- Directly storing data in JanusGraph database (on top of HBase)
- Accessing them via standard Gremlin interface of the TinkerPop framework
- Prototype 2 - selected for Run 3
  - Storing data in HBase tables with Phoenix SQL API
  - To allow interoperability with other SQL-based ATLAS services
  - With additional HBase tables adding (lazy) graph structure on top
  - Very generic Dynamical Web Service GUI
  - In principle re-usable for any Graph-like data
  - Customizable via stylesheets

The global dynamical interactive view of all ATLAS data
- With relations between data entities
- Giving access to all available services
- User-extensible
- Usable for HEP analytics

See also: Presentation Using Graph Databases in HEP (Track 4, Th, 15:15)