Object-oriented databasing for nuclear data

Adam Hayes, Benjamin Shu, Libby McCutchan,
Shaofei Zhu, Alejandro Sonzogni

National Nuclear Data Center
Brookhaven National Laboratory

WPEC SG50
14 Sep 2020

Let’s talk about...

- Current data storage at NNDC
- Preparing for the future
- Overview of contemporary storage systems
- NNDC test-case for upgraded storage

ENSDF output

Coulomb excitation 2005Zh20,1957Ne07 (continued)

^{239}Pu Levels (continued)

E(level)†	Jπ	E(level)†	Jπ	E(level)†	Jπ	E(level)†	Jπ
1847.0@ *15*	35/2+	2529.4a *23*	39/2−	3108.0& *20*	45/2−	3895a *3*	51/2−
1908.9& *15*	33/2−	2590.1# *17*	41/2+	3198.0@ *22*	47/2+	4080.0& *24*	(53/2−)
2143.4a *21*	35/2−	2672.0& *17*	41/2−	3407a *3*	47/2−	4087.1# *24*	(53/2+)
2152.2# *16*	37/2+	2714.0@ *19*	43/2+	3559.1# *22*	(49/2+)	4256@ *3*	(55/2+)
2263.0@ *16*	39/2+	2951.4a *25*	43/2−	3578.0& *22*	(49/2−)	4413a *3*	55/2−
2272.0& *16*	37/2−	3060.1# *20*	45/2+	3713.0@ *24*	(51/2+)		

† Deduced by evaluators from least-squares fit to Eγ's; ΔEγ=1 keV assumed for each transition, unless otherwise noted.
‡ From ^{239}Pu in Adopted Gammas.
# Band(A): 1/2[631], α=+1/2.
@ Band(a): 1/2[631], α=−1/2.
& Band(B): Octupole band, α=+1/2. Band associated with octupole vibration at low spin.
a Band(b): Octupole band, α=−1/2. Band associated with octupole vibration at low spin.

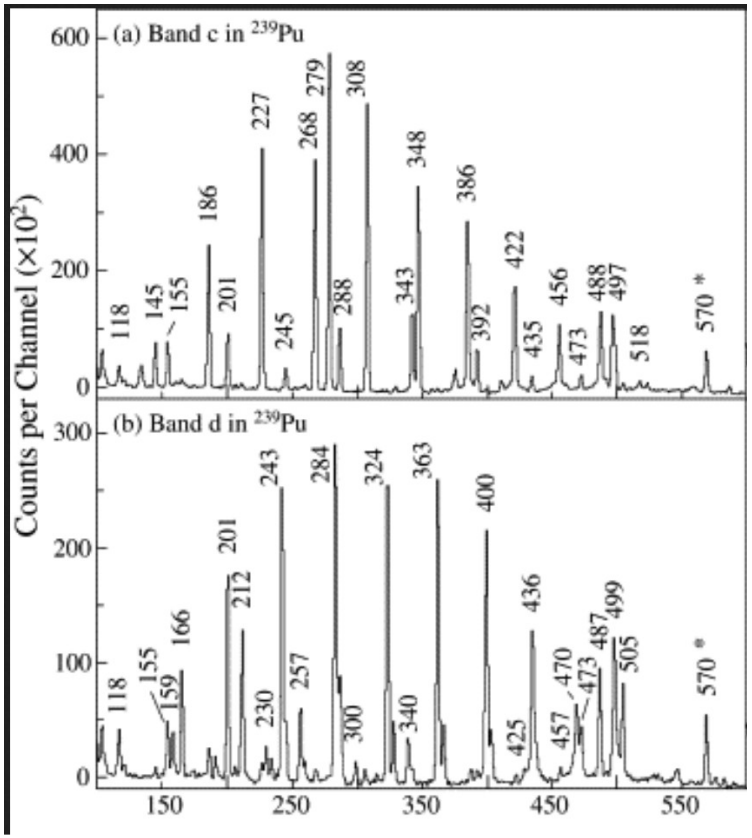
γ(^{239}Pu)

Eγ	Ei(level)	Jiπ	Ef	Jfπ	Mult.§
7.860† *3*	7.861	3/2+	0.0	1/2+	
49.412† *4*	57.275	5/2+	7.861	3/2+	M1+E2‡
57.273† *4*	57.275	5/2+	0.0	1/2+	E2
67.841† *7*	75.705	7/2+	7.861	3/2+	E2

ENSDF storage

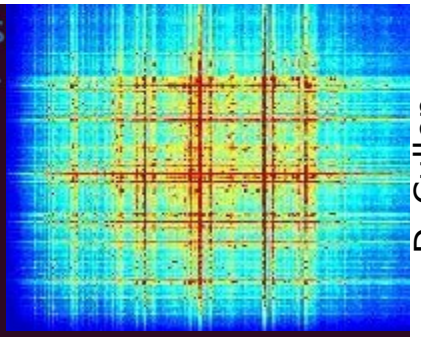
```
239PU cG M(Z)$|g|g(|q) analysis supports stretched E2 transition (2005Zh02)
239PU cG M(T)$Linking transition suggested by 2005Zh20 as E1
239PU cG M(E)$From |g-ray angular distributions (1993De12)
239PU cL E$Deduced by evaluators from least-squares fit to E|g's; |DE|g=1 keV
239PU2cL assumed for each transition, unless otherwise noted.
239PU cL E(Y)$From {+239}Pu in Adopted Gammas.
239PU CL BAND(A)$ 1/2[631], |a=+1/2
239PU CL BAND(a)$ 1/2[631], |a=-1/2
239PU CL BAND(B)$ OCTUPOLE BAND, |a=+1/2.
239PU2cL Band associated with octupole vibration at low spin.
239PU CL BAND(b)$ OCTUPOLE BAND, |a=-1/2.
239PU2cL Band associated with octupole vibration at low spin.
239PU  L 0.0          1/2+                                                  A
239PU  L 7.861      2 3/2+                                                  a
239PU DL E          LEVEL ENERGY HELD FIXED IN LEAST-SQUARES ADJUSTMENT
239PUF L FLAG=Y
239PU  G 7.860      3                                                       X
239PU  L 57.275     2 5/2+                                                  A
239PU DL E          LEVEL ENERGY HELD FIXED IN LEAST-SQUARES ADJUSTMENT
239PUF L FLAG=Y
239PU cL            B(E2)=5.3 {I3} (1957Ne07) using |a=214
239PU  G 49.412     4           M1+E2                                       X
239PU2 G FLAG=Y
239PU  G 57.273     4           E2                                          X
239PU  L 75.705     3 7/2+                                                  a
239PU DL E          LEVEL ENERGY HELD FIXED IN LEAST-SQUARES ADJUSTMENT
```

What if?



S. Zhu et al., Phys Lett B 618 51 (2005)

```
/* process good events
/*--------------------

if (st == GOODEV)
{


  /* event time  */
  /* take v.r.t user specified t0 */

  EvTime = hdr.ttL + 65536.0 * hdr.ttM
  EvTime /= 1000000.0;>-----/* now seco
  EvTime -= EvTime0;

  /* update event rate spectrum */

  if (EvTime > 0 && EvTime < LENEVTIME)
----evrate->Fill(EvTime);
```



D. Cullen

```
00000740  51 f1 c4 8e db 8d fd 94  1b ce db 38 da 19 4f e6  |Q..........8..O.|
00000750  f4 de e1 af ef f4 55 ec  2e 18 aa 0c 1b b9 2f d5  |......U......./.|
00000760  60 7e 6e 03 d1 3d 7f 71  3f 4c 67 d7 6f 17 e1 06  |`~n..=.q?Lg.o...|
00000770  6c 5b 92 c9 1f 70 49 c8  bd d9 14 fd 77 37 74 da  |l[...pI.....w7t.|
```

How to store big(ger) data: Guiding concepts

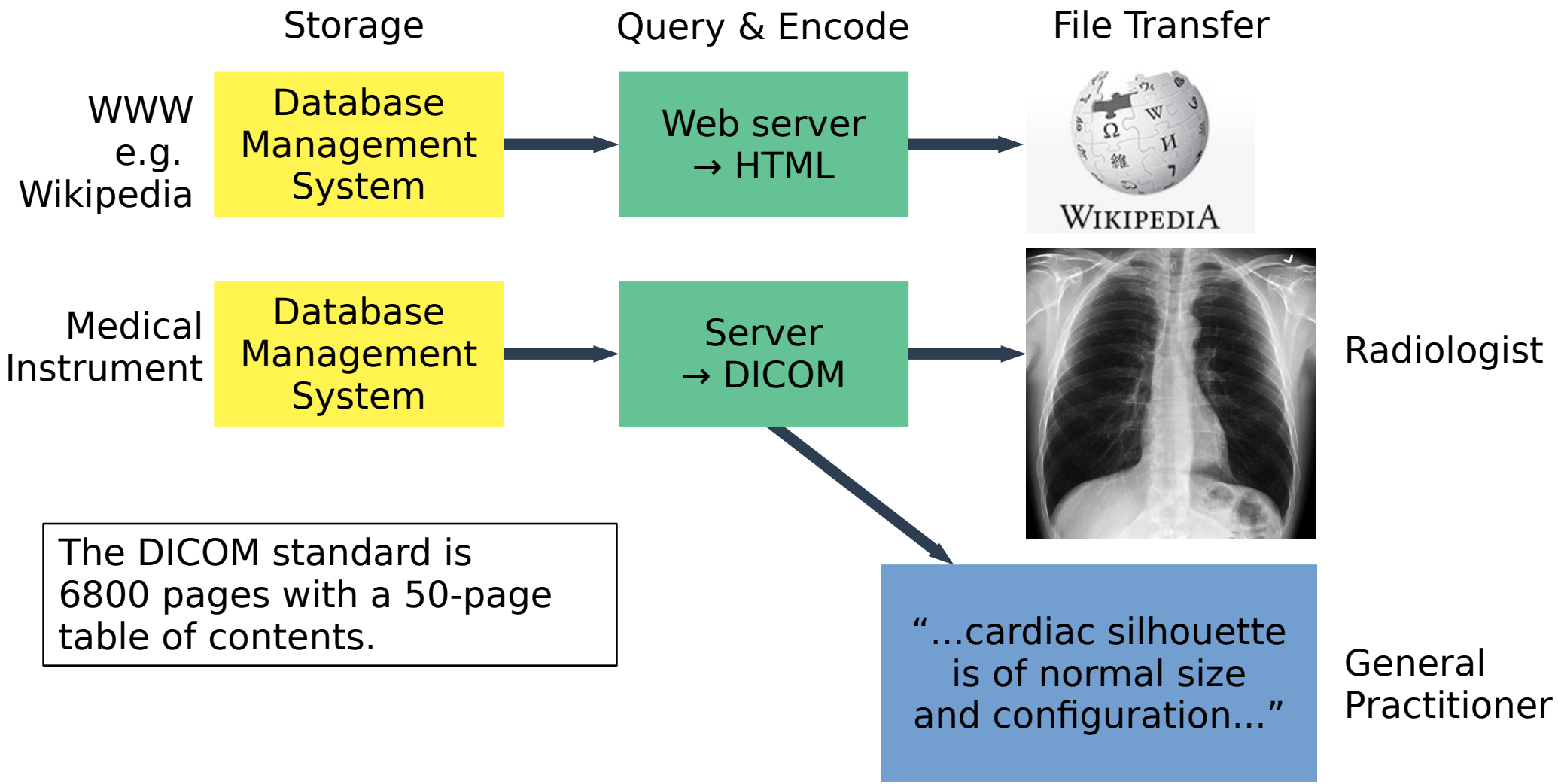
- Start with contemporary technology, be prepared for
  - Unknown data in the future
  - Our heirs
- *Internally*, use a database; *export* file for transmission
- Keeping concepts of storage and transmission separate benefits the community and facilitates community contributions to both.

File formats vs. storage

Think *transmission* (interchange) when we think about file formats; think *storage* as separate from a particular file format.

- File formats
  - Take time, money, and collaboration to update
  - “Locked in” for some period thereafter
  - Different users may benefit from different formats
  - Codes using an older format may be difficult to update
- Storage
  - Can change internal storage (database) without changing file format
  - Can be a black box
  - Curating more complex data in the future: prepare for binary blobs that can be excluded from particular file formats

File formats vs. storage



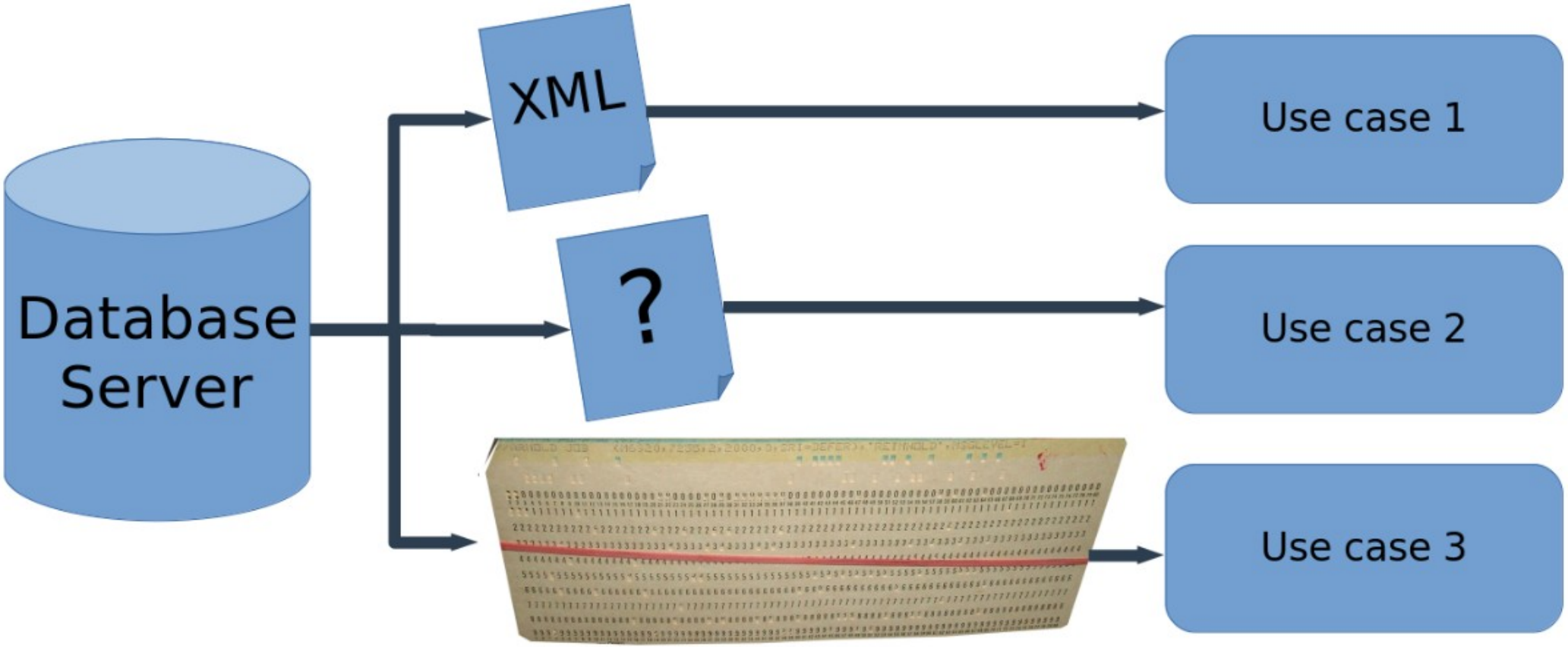
When storage and transmission are separated…

- Users of data can be **agnostic** of the storage system.
- The storage system can be changed, upgraded ***independently*** of the file format.
- Additions to the types of stored data can be implemented swiftly.
- The storage system can be expandable.
- Standard API is used to access data.
- Don’t need to reinvent the wheel (standard database functions).
- If you add to file format, everyone has to deal with it;
  you can add something new to a database unobtrusively.

Storage vs. transmission: accomodating multiple end-users

Storage

Transmission (file format)



Step back: Brief history of databases

(Focusing on relational / object-oriented)

- 1960s – 1970s: advent of relational databases, origins of modern types
- 1980s: SQL (Dbase III)
  experiments with Object-oriented Databases
- 1990s: remote servers more common
- 1995-6: MySQL, PostgreSQL (relational)
- 2000s: CouchDB, MongoDB (object oriented)

  .

  .

  .
- 2020: Big data, cloud, replication, many variants of common database types...

Relational vs. Object-oriented DBs

Relational databases (RDBMS)

100	22
101	28
102	33
103	42
104	31

- “Tables, rows, columns, references”
- In use for ENSDF, NSR, etc.
- e.g. MariaDB, MySQL…
- Originally created to handle simple data types
- Quickly becomes mess for complex data types (dictionaries, mixed types, dynamic arrays…)
- Complex types require references between tables

→ schema design and lookups become complicated

Relational vs. Object-oriented DBs

Object-oriented databases (OODBMS)

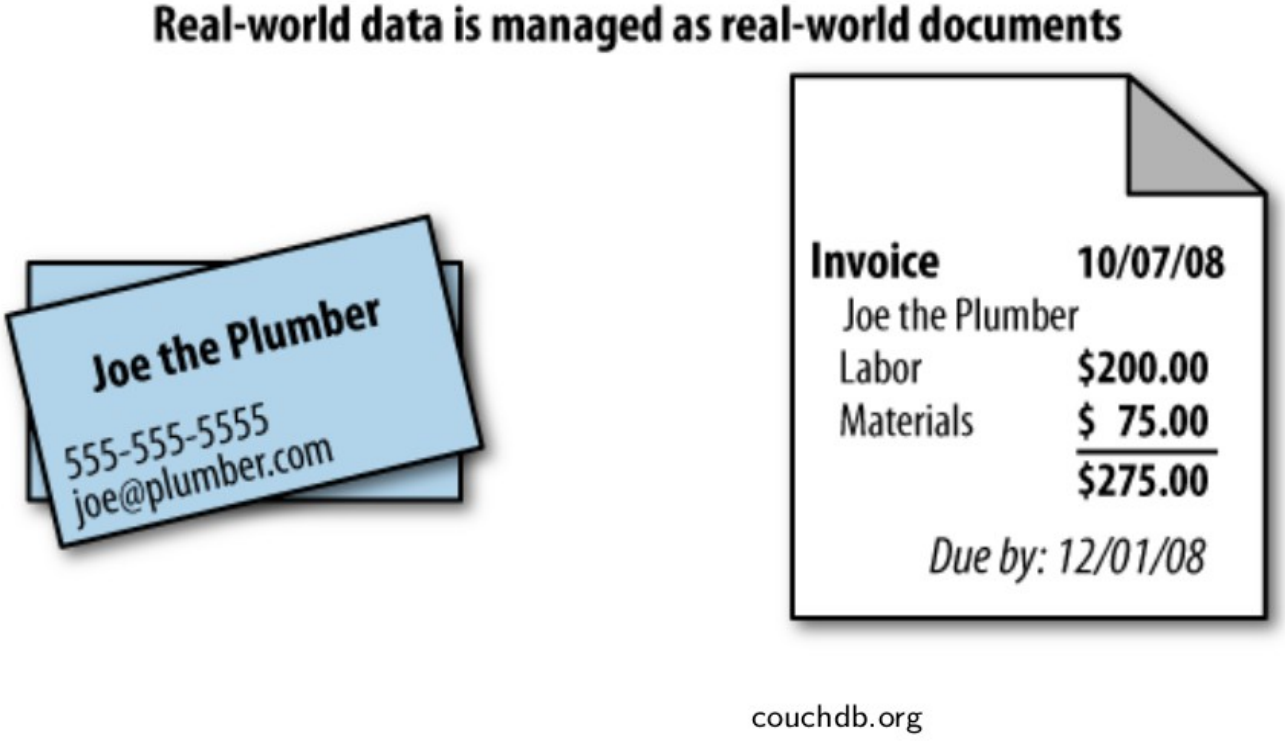
- “Documents (record), collections”
- Maps to object-oriented code more naturally
  - Akin to storing objects themselves (“documents”)
  - Less work reading data into (code) objects
- e.g. MongoDB, CouchDB
- Hierarchical structure, but easily changed “on the fly” (rapid development cycle, large teams)
- Binary data can be inside documents
- Complex or dynamic types (lists, dictionaries/maps)
- Search complex document by fairly easily

Why object-oriented database?

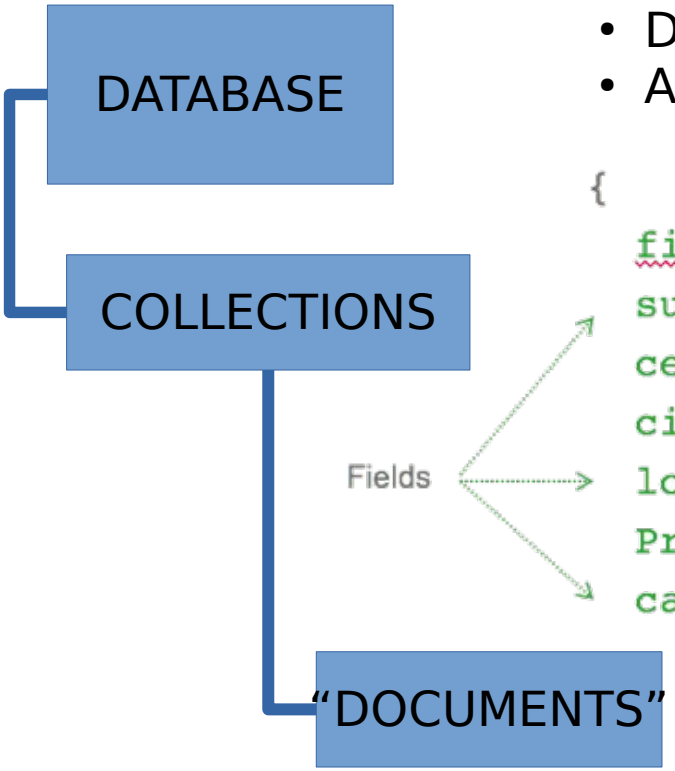
- Variety of data types
  - Float, int, etc.
  - List, dict, etc.
  - Long-form text
  - Documents
  - Images (e.g. plots)
  - Binary data
- Expansible without disruptive changes to format
- Hierarchical documents (records) with fewer cross-references
- Simplicity paradigm: “Store together what you access together”

Object-oriented database “documents”

Think “records”



Object-oriented database “documents”



- Document structure; **NOT** stored as a file
- Accessed via database query / search

```
{
  first_name: ‘Paul’,
  surname: ‘Miller’,
  cell: 447557505611,
  city: ‘London’,
  location: [45.123,47.232],
  Profession: [‘banking’, ‘finance’, ‘trader’],
  cars: [
    { model: ‘Bentley’,
      year: 1973,
      value: 100000, … },
    { model: ‘Rolls Royce’,
      year: 1965,
      value: 330000, … }
```

Fields

String

Number

Geo-Coordinates

Typed field values

Fields can contain arrays

Fields can contain an array of sub-documents

MongoDB.com

“Views”

Good example of not reinventing the wheel

> “Views are useful for many purposes:
>
> - Filtering the documents in your database to find those relevant to a particular process.
> - Extracting data from your documents and presenting it in a specific order.
> - Building efficient indexes to find documents by any value or structure that resides in them.
> - Use these indexes to represent relationships among documents.
> - Finally, with views you can make all sorts of calculations [*e.g. quantity of data for a reaction*].
>
> ”
>
> —couchdb.org

Comparison of two popular OODBMSs

- **MongoDB** (2009)
  - eBay, Google, Facebook, PayPal, CERN
  - Future license terms uncertain
  - May be more version compatibility issues between API and server
  - Uses “collections” of similar-type documents in a database
  - Database → Collections → Documents
  - Updating a document is allowed
  - Massively scalable
  - “GridFS” for huge blobs
  - API

Comparison of two popular OODBMSs

- **CouchDB** (2005)
  - Apple, GrubHub, Credit Suisse, Motorola, some Facebook Apps
  - Permissive, irrevocable Apache license
  - No concept of “collections”; all document types stored together
  - Database → Documents (with “type” as a field)
  - Updating not allowed; modify document and (re)insert
  - Massively scalable
  - Can store large (configurable) blobs or link to outside storage for huge blobs
  - API / simple http communication

Test case: Nuclear Wallet Cards

NUCLEAR WALLET CARDS

Jagdish K. Tuli

National Nuclear Data Center
www.nndc.bnl.gov

Brookhaven National Laboratory
P.O. Box 5000
Upton, New York 11973-5000
U.S.A.

Nuclear Wallet Cards

Nuclide Z	El	A	Jπ	Δ (MeV)	T½, Γ, or Abundance	Decay Mode
92	U	221	(9/2+)	24.6s	700 ns	
		222	0+	24.3s	1.0 μs *+12–4*	α
		223		25.84	18 μs *+10–5*	α, ε 0.2%
		224	0+	25.71	0.9 ms *3*	α
		225		27.38	95 ms *15*	α
		226	0+	27.33	0.35 s *15*	α
		227	(3/2+)	29.02	1.1 m *1*	α
		228	0+	29.22	9.1 m *2*	α>95%, ε<5%
		229	(3/2+)	31.209	58 m *3*	ε≈80%, α≈20%
		230	0+	31.613	20.8 d	α, SF<1×10^{-10}%, ^{22}Ne 5×10^{-12}%
		231	(5/2–)	33.807	4.2 d *1*	ε, α≈4.0×10^{-3}%
		232	0+	34.604	68.9 y *4*	α, SF 3×10^{-12}%
		233	5/2+	36.921	1.592×10^{5} y *2*	α, ^{24}Ne 9×10^{-10}%, SF<6×10^{-11}%, ^{28}Mg<1.×10^{-13}%
		234	0+	38.148	2.455×10^{5} y *6* **0.0054% *5***	α, SF 1.6×10^{-9}%, Mg 1×10^{-11}%, Ne 9×10^{-12}%
		235	7/2–	40.921	7.04×10^{8} y *1* **0.7204% *6***	α, SF 7.0×10^{-9}%, ^{28}Mg 8.×10^{-10}%, Ne ≈8.×10^{-10}%
		235m	1/2+	40.921	≈26 m	IT
		236	0+	42.447	2.342×10^{7} y *4*	α, SF 9.4×10^{-8}%
		237	1/2+	45.393	6.75 d *1*	β–
		238	0+	47.310	4.468×10^{9} y *3* **99.2742% *10***	α, SF 5.5×10^{-5}%
		239	5/2+	50.575	23.45 m *2*	β–
		240	0+	52.716	14.1 h *1*	β–
		241		56.2s		β–?
		242	0+	58.6s	16.8 m *5*	β–
		243		62.4s		

WalletCraft database and UI: Motivation

- Need system to speed & streamline updates of Nuclear Wallet Cards
- Synchronize ENSDF evaluation & Wallet Cards updates
- Uniform calculation of mean half-lives, etc.
- Store
  - Source data: measured values from publications
  - Evaluated quantities
  - Comments, notes, etc. separated from numerical data; all numerical data in numerical fields
- Version tracking
  - Freeze “evaluation” versions
  - Tag “publication” versions
- Shorten time to publish a version
- **Test case for future upgrades to larger databases**

Planning ahead for expansion

- "Everything is a dict"
- Dict = map ~ JSON, but can store *objects in database*
- Expand "schema" without need to change client, e.g.
  - **OK**: (Hard to maintain, disruptive changes)
    `{"key":value}`
    Need to change data type to expand:
    `{"key":[value1,value2]}`
  - **BETTER**: (Expandible, but confusing)
    `{"key":[value1,value2,…]}`
  - **BEST**: (Expand without disrupting users & developers)
    `{"key":{"value1Name":{"attribute1":value1}}}`

    Easily add an attribute:
    `{"key":{"value1Name":`
    `{"attribute1":value1,"attribute2":value2…}}}`

Database “schema”
Full nuclide entry


```
{
    "_id": "62,163",
    "_rev": "1-4b5a0b7904644dab711ce77c6cbb20ea",
    "Z": 62,
    "A": 163,
    "symbol": "163Sm",
    "levels": {↔},
    "rawLines": ["163  SM   62   101  201704    0.0
    "documentType": "nuclide",
    "debug": "Inserted with add-ensdf-to-wc.py"
}
```

Database “schema”
Levels section


```
{
    "_id": "62,163",
    "_rev": "1-4b5a0b7904644dab711ce77c6cbb20ea",
    "Z": 62,
    "A": 163,
    "symbol": "163Sm",
    "levels": {
        "GS": {↔}
    },
    "rawLines": ["163  SM   62   101  201704    0.0
    "documentType": "nuclide",
    "debug": "Inserted with add-ensdf-to-wc.py"
}
```

Database “schema”
Ground-state level

```
"levels": {
    "GS": {
        "tHalf": {↔},
        "decayWidth": {↔},
        "massExcess": {↔},
        "abundance": {↔},
        "decayModes": {↔},
        "Jpi": {↔},
        "energy": {↔}
    }
},
```

Database “schema”
Half-life of ground state

```
4    "Z": 62,
5    "A": 163,
6    "symbol": "163Sm",
7    "levels": {
8        "GS": {
9            "tHalf": {
10               "published": {},
11               "evaluations": {↔},
27               "measurements": {}
28           },
```

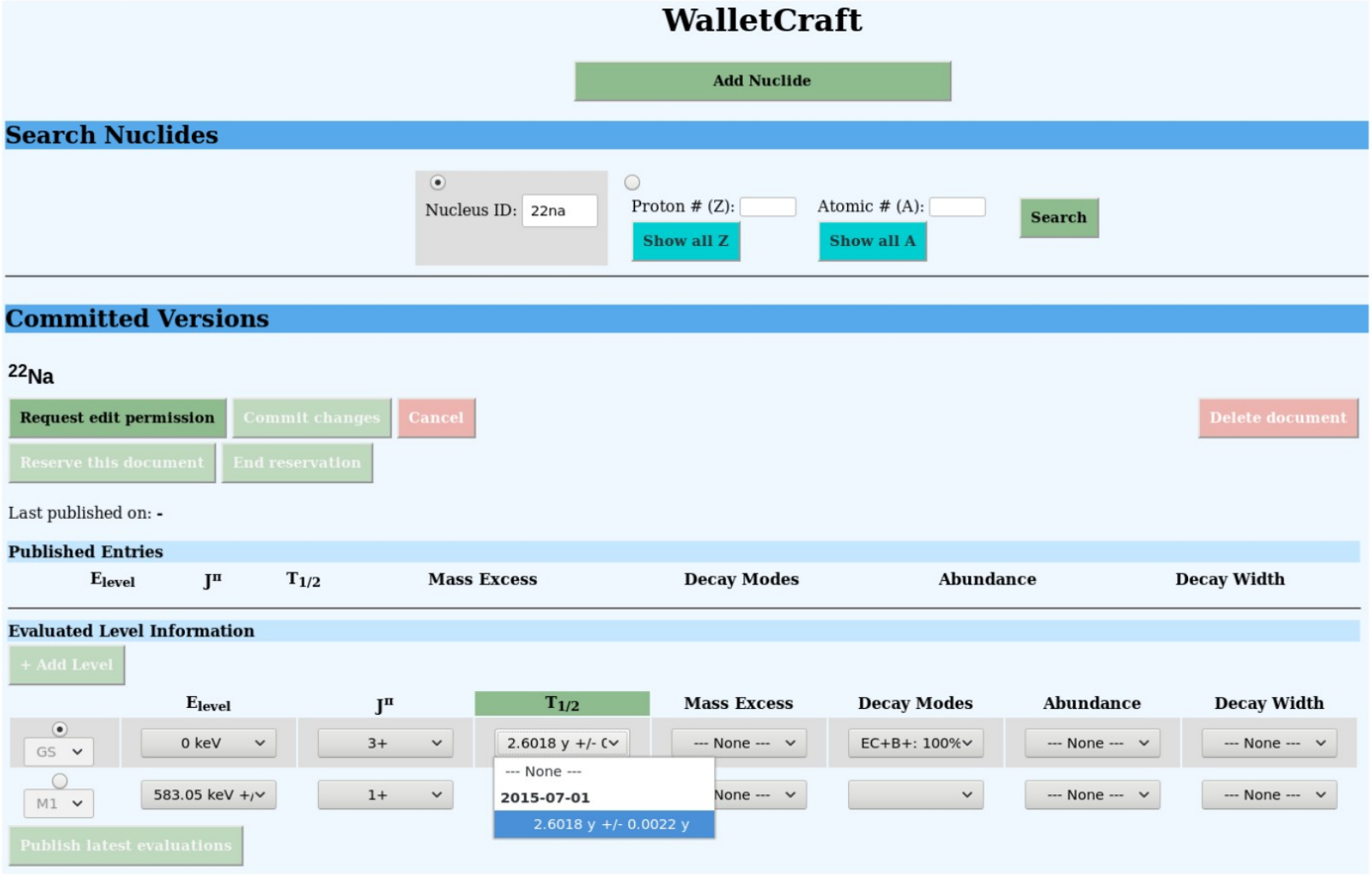
Database “schema” Evaluations

```
    "Z": 62,
    "A": 163,
    "symbol": "163Sm",
    "levels": {
        "GS": {
            "tHalf": {
                "published": {},
                "evaluations": {
                    "2017, 04, 01, 00, 00, 00, 000000": {↔}
                },
                "measurements": {}
            },
```

Database “schema”
Individual evaluation (example only)


```
"evaluations": {
    "2017, 04, 01, 00, 00, 00, 000000": {
        "tHalfIsLowerLimit": false,
        "tHalfIsUpperLimit": false,
        "tHalfIsUpperEqual": false,
        "tHalfIsLowerEqual": false,
        "tHalfIsApproximate": false,
        "tHalfIsUnknown": false,
        "tHalf": 1.23,
        "tHalfUnit": "s",
        "tHalfInSeconds": 1.23,
        "dtHalf": [0.51, 0.47],
        "stable": false,
        "measurementsCited": {}
    }
```

WalletCraft UI (Ben Shu)



Ben Shu (NNDC)

WalletCraft UI (Ben Shu)

NSR Key	$T_{1/2}$ & Δs				Experimental Technique	Comments	
2034Sc04	32.3	+ 0.4	- 0.4	h			X
2030Un02	30.1	+ 0.3	- 1.0	h			X
2025Re12	54	+ 3	- 3	h		Reason for exclusion...	X
1342Am15	33	+ 5	- 5	h			X

+ Add measurement

Calculations

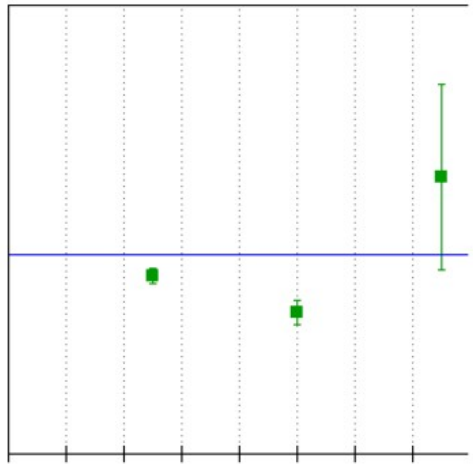
Type: Weighted Average

Evaluated Value: 32.1913 h
+ 32.191341
- 32.191341

Add comment: Weighted average was used because...

Re-calculate

Save evaluation



WalletCraft Status

- ~3000 nuclide documents
  - Mostly basic information ($t_{1/2}$, J^{π}, etc.)
- Version history
- Tracking of
  - Evaluations
  - Published versions
- UI and server in internal use
- Will output original Wallet Cards file format (for now)
- Evaluators will use WalletCraft for the next version of Wallet Cards (2020-2021)

Summary

- Keeping concepts of storage and transmission separate
  - benefits the community
  - facilitates community contributions to both (parallel development)
- [I hate change, but] **change is** [often] **good**
- New data storage paradigms:

  prepare for the *next* 50 years of nuclear data

Clipart: openclipart.org/artist/j4p4n and espertus

Collaborators on WalletCraft:

Benjamin Shu, Libby McCutchan, Shaofei Zhu, Alejandro Sonzogni

END