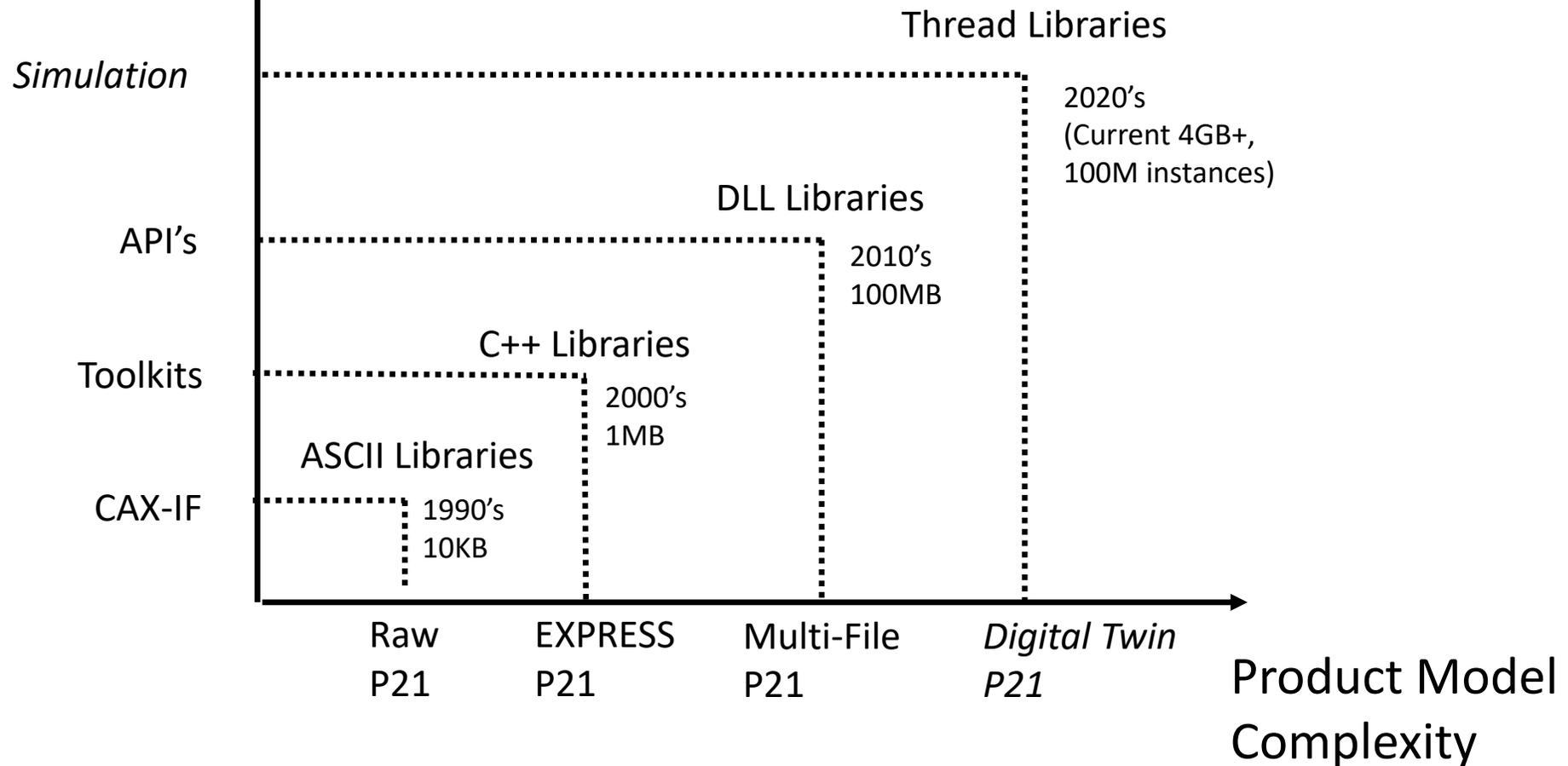# Digital Twin Part 21

Martin Hardwick

WG11 & WG15

Background

- Part 21 has been the primary data format for STEP since the 1990's.

- Digital twins will require product models of great size and complexity.

- Twins will "challenge" applications because there will be too many entities for reasonable processing.

- For 10 years STEP Tools has been using an informal extension to Part 21 to help it make sense of long complex product models.

- Maybe it is time to make this informal extension a formal one.

Evolution of Part 21

Product Model
Functionality

Raw P21: Read and write using parser
EXPRESS P21: Read and write using compiler
Multi-File P21: Read and write subsets

Thread Libraries

Simulation

2020's
(Current 4GB+,
100M instances)

DLL Libraries

API's

2010's
100MB

C++ Libraries

Toolkits

2000's
1MB

ASCII Libraries

CAX-IF

1990's
10KB

Raw
P21

EXPRESS
P21

Multi-File
P21

*Digital Twin
P21*

Product Model
Complexity

Example

- This is the P21 used by STEP Tools

- Catalog/index data is optional and in comments

- The catalog data is a "Dewey Decimal Index" for STEP

- The proposal is to move the catalog out of the comments and into the Part 21 standard

External identifiers

Catalog data

AIM data

```
STEP File Browser - Simple_program_wtih_tool_assembly.stpnc [page 1/34]
File   View   Navigate   Help

ANCHOR;
<1727b5d4-dbe7-4989-8493-68aface34eb8>=#624;   /* line 29 WS 2 TP 1 */
<12677aa7-7ac2-4dc8-a047-72712798d98c>=#633;   /* line 35 WS 2 TP 2 */
<32f9c5e3-2b31-427c-a672-f271da5cf82e>=#678;   /* line 115 WS 2 TP 3 */
<0295d603-ecd3-4648-8e97-c59ed8cd0053>=#690;   /* line 123 WS 2 TP 4 */
<ae8952bd-c0a3-402c-a835-295b744a1923>=#697;   /* line 127 WS 2 TP 5 */
<bf45a370-3ce8-4f86-9495-5666399ca765>=#742;   /* line 207 WS 2 TP 6 */
ENDSEC;

DATA;

/****************************************************************
 * Application object: PROJECT (#10)
 * MAIN_WORKPLAN: #10, #11, #12, #938
 * ITS_WORKPIECES [*]: #10, #13, #265
 * ITS_ID: #10, #14, #15, ['Simple_program_wtih_tool_assembly']
 */
#10=PRODUCT_DEFINITION('','',#14,#16);
#11=PROCESS_PRODUCT_ASSOCIATION('','',#10,#12);
#12=PRODUCT_DEFINITION_PROCESS('machining','',#938,'');
#13=MACHINING_PROJECT_WORKPIECE_RELATIONSHIP('','','',#10,#265);
#14=PRODUCT_DEFINITION_FORMATION('','',#15);
#15=MACHINING_PROJECT('Simple_program_wtih_tool_assembly','',$,#17
#16=PRODUCT_DEFINITION_CONTEXT('CNC Machining',$,'manufacturing');
#17=PRODUCT_CONTEXT('CNC Machining',$,'manufacturing');

/****************************************************************
 * Application object: WORKPIECE (#18)
 * ITS_RELATED_GEOMETRY [*]: #18, #19, #20, #21, #22, #23
 * ITS_CATEGORIES [*]: #18, #24, #25, #26, ['part']
 * ITS_GEOMETRY: #18, #19, #20, #21
 * ITS_STYLED_MODELS [*]: #18, #27, #28, #949
 * ITS_STYLED_MODELS [*]: #18, #29, #30, #950
 * SHAPE_DEFINITION: #18, #19
 * ITS_ID: #18, #24, #25, ['PART1']
 * ITS_CONSTRUCTIVE_MODELS [*]: #18, #19, #20, #21, #31, #1017
 * REVISION_ID: #18, #24, ['']
 */
#18=PRODUCT_DEFINITION('',' ',#24,#32);
#19=PRODUCT_DEFINITION_SHAPE(' ',' ',#18);
#20=SHAPE_DEFINITION_REPRESENTATION(#19,#21);
#21=SHAPE_REPRESENTATION(' ',(#34,#923),#39);
#22=SHAPE_REPRESENTATION_RELATIONSHIP(' ',' ',#21,#23);
#23=ADVANCED_BREP_SHAPE_REPRESENTATION('NONE',(#963),#39);
#24=PRODUCT_DEFINITION_FORMATION_WITH_SPECIFIED_SOURCE('',' ',#25,
.NOT_KNOWN.);
#25=PRODUCT('PART1','','',(#264));
#26=PRODUCT_RELATED_PRODUCT_CATEGORY('part',$,(#25));
#27=PROPERTY_DEFINITION('styled model','',#18);
#28=PROPERTY_DEFINITION_REPRESENTATION(#27,#949);
#29=PROPERTY_DEFINITION('styled model','',#18);
#30=PROPERTY_DEFINITION_REPRESENTATION(#29,#950);
#31=CONSTRUCTIVE_GEOMETRY_REPRESENTATION_RELATIONSHIP(
'supplemental geometry','',#21,#1017);
#32=PRODUCT_DEFINITION_CONTEXT('part definition',#33,' ');
#33=APPLICATION_CONTEXT(
'configuration controlled 3D design of mechanical parts and assembl
#34=AXIS2_PLACEMENT_3D(' ',#35,$,$);
#35=CARTESIAN_POINT(' ',(0.,0.,0.));
#36=CARTESIAN_POINT('',(0.,0.,0.));
#37=DIRECTION('',(0.,0.,1.));
#38=DIRECTION('X direction',(1.,0.,0.));
```

Some requirements

- Make STEP data easier to understand and debug
  - To enable more complex digital twins
- Support multiple classification/index systems
  - Built from mapping tables, EXPRESS-X, SysML and other languages
- Manage seamless upgrade from current P21
  - So catalogs and indexes can be added to existing data
- Enable lightweight access
  - So simple applications can be written at lower cost
- Support multiple data formats
  - So QIF results can explain STEP measurements

More requirements

- Support very large product models
  - For example a digital twin of an airframe
- Enable multiplicity of signatures
  - Divide data into indexed sections that can be signed individually

Scalable and secure
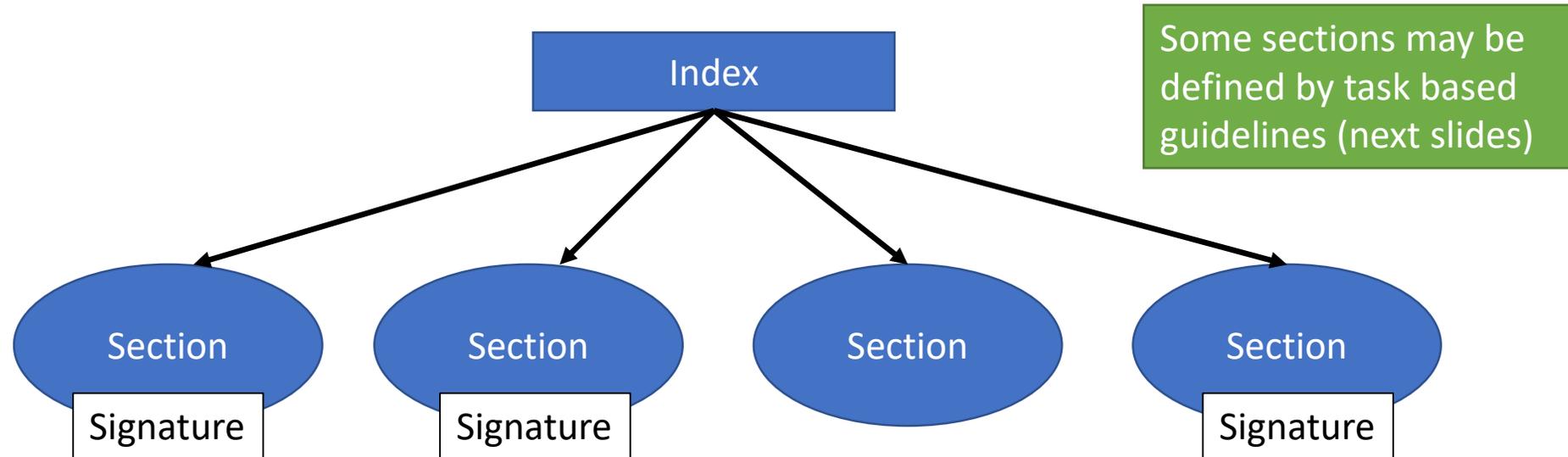
- Sum of sections is current P21
- Each section may have its own signature
- Section may be a product with documents (pdwss) to references external source
- Index entries can be generic (as per STEP Tools) or specific as per following slides

Context-Based Machine Monitoring/State Display

"Surely I should know more about what is going on than a bunch of numbers changing on a screen"

- This index makes a machining project

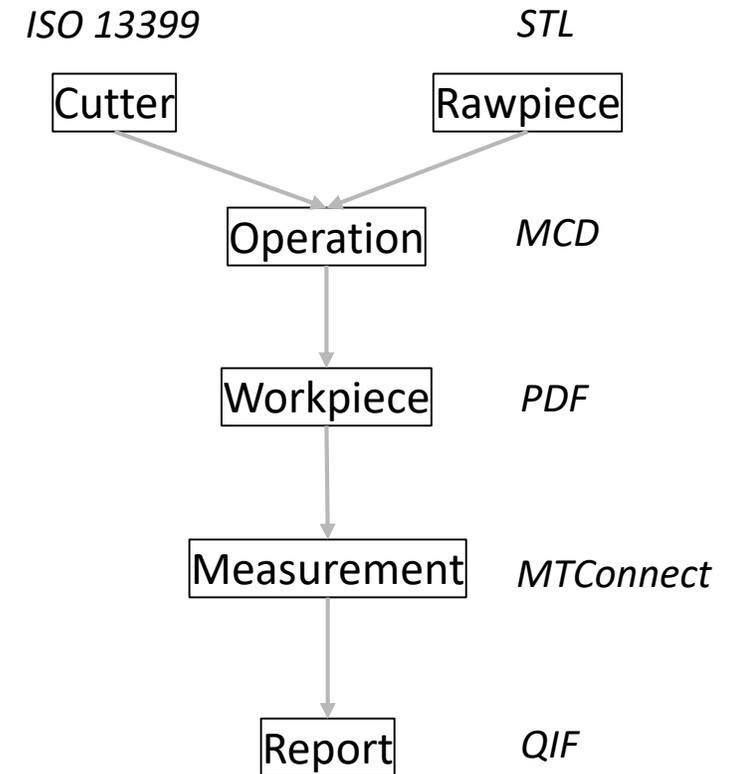- The index can be read directly

- The index converts to P21 e2

ENTRY ⟶ PDWSS ⟶ P21

- ISO 23247 defines the conversion?

```
INDEX context_based_machining
USE FROM file_reference

ENTRY project;
   name : STRING;
   operations: LIST [1:?] OF operation;
   measurements: OPTIONAL file_reference;
   report: OPTIONAL file_reference;
END_ENTRY;

ENTRY operation
   name: STRING
   cutter: file_reference
   rawpiece: file_reference
   instructions: file_reference
   workpiece: file_reference
END_ENTRY;
END_INDEX;
```

ISO 13399   *STL*

Cutter   Rawpiece

Operation   *MCD*

Workpiece   *PDF*

Measurement   *MTConnect*

Report   *QIF*

Full Models Available Pre-Purchase

"Why is it so hard to buy a machine?"

- The index defines how to place a project on a machine to run a simulation test

- The fixture is an optional addition and must be in the same coordinate space as the project

```
INDEX digital_machine_test
USE FROM context_based_machining

ENTRY machine_test;
    name : STRING;
    machine: file_name;
    kinematics: OPTIONAL file_reference;
    tests: LIST [1:?] of simulation
END_ENTRY;

ENTRY simulation
    name: STRING
    date: date_and_time
    fixture: file_reference
    project: project
  placement: OPTIONAL axis2_placement_3d
END_ENTRY;
END_INDEX;
```

Machine   *STEP*

Kinematics   *XML*

    STL
Project ← Fixture

Report   *QIF*
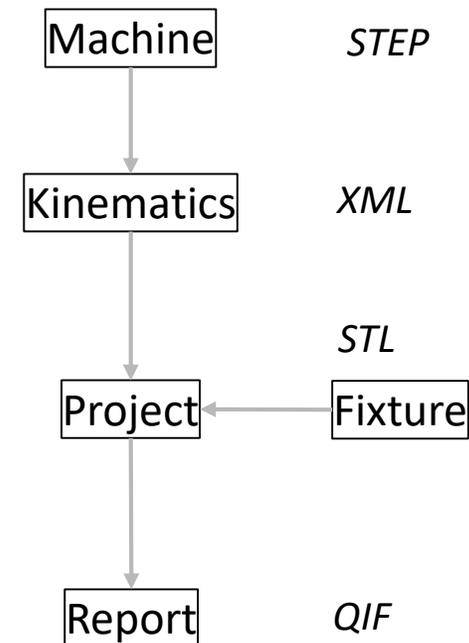
Security/Integrity/Provenance

"How do I know that I am machining with the right program"

- We need to assure files are ready for machining
  - As defined by the customer
  - Not altered by third parties
- Check sums validate the data has not been altered
- Signatures validate the identity of the author
  - Who signed
  - For what purpose
- See Embedding X.509 Digital Certificates in Three-Dimensional Models for Authentication, Authorization, and Traceability of Product Data, Hedburg, Krima and Camelio, JCISE Vol 17, March 2017

```
INDEX file_reference
REFERENCE pks_x509

ENTRY file_reference;
  name : file_name;
  file_type: STRING;
  check_sum: INTEGER;
  signature: LIST OF pks_trace;
END_ENTRY;

TYPE file_name = STRING;
  WHERE
    wr1: scope_is_local ();
END_TYPE;
END_INDEX;
```

ZIP Archive

Index

STL

13399

PDF

QIF
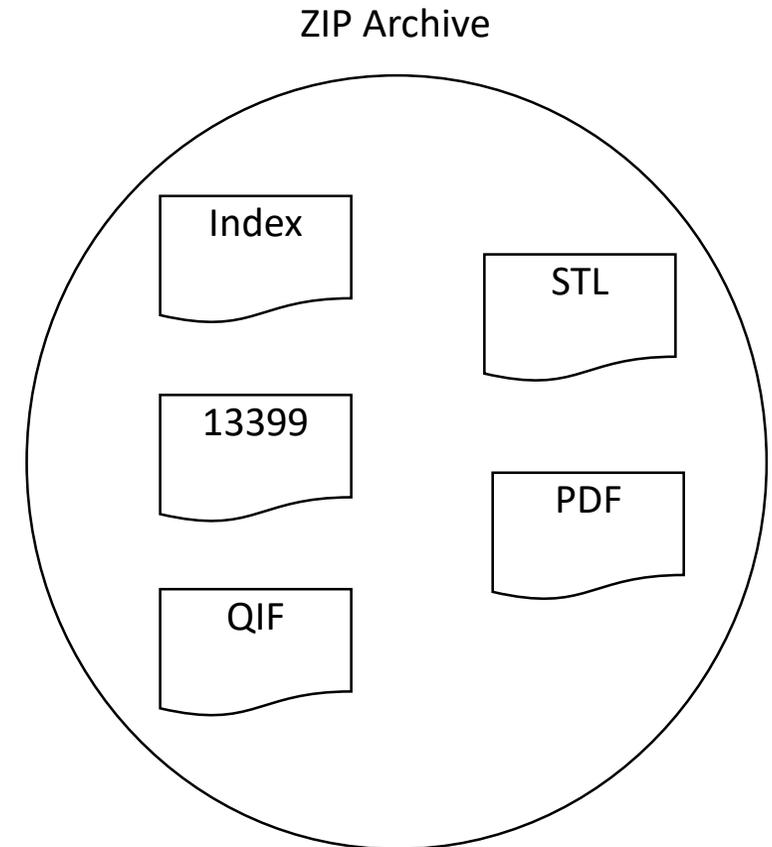
Why P21

"XML and JSON are much more widely used"

- Can develop for the requirements of the product modeling community
- Have an effective support infrastructure
  - CAX-IF, PDM-IF, DM-IF and multiple vendors
- Many of the necessary features are already in Part 21 Edition 3
  - UUID's
  - Multiple data sections
  - Signatures and check sums
  - URL references

Issues to think about

- How to encourage participation in the testing and development
- Should this be a 10303 standard or a 23247 standard
- How many types of indexes should be supported
  - Just a generic one, or also "funky" application specific ones
- What should be signed?
  - The index entry, the data section or both?
- What should be the timeline for completing the standard