A Language for Ultra-Long-Term CAD Data Preservation

Walter W. Wilson
Lockheed Martin
Fort Worth, Texas

SIAM Conf. on Geometric Design 2019

www.axiomaticlanguage.org/GD19_slides.pdf

Outline

- Motivation for ultra-long-term preservation
- Principles for ultra-long-term CAD preservation
- A language for ultra-long-term preservation
- A vision for CAD

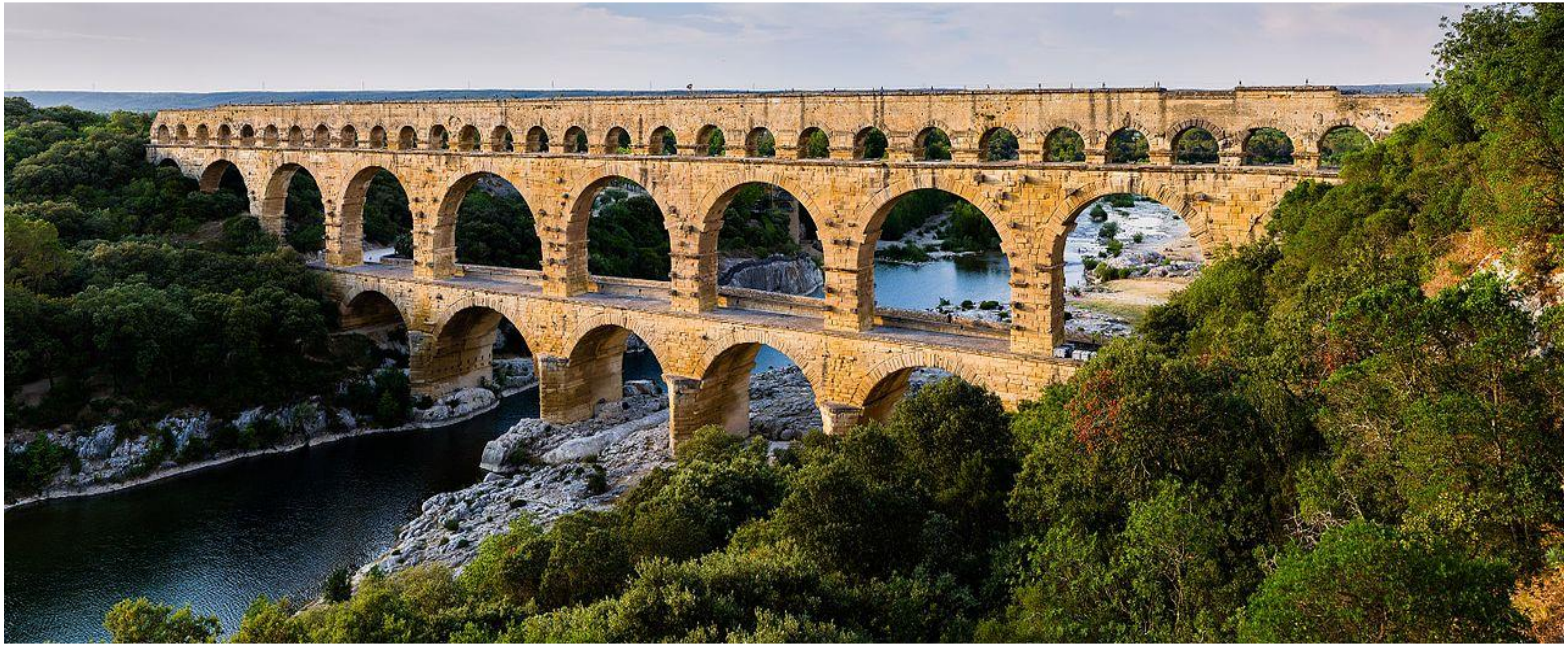
Airplanes can last for decades, maybe a century …

Ford Trimotor (1925)

May want to save historic data forever …

North American X-15 (1959-1968)

Buildings and other structures may last thousands of years …



Pont du Gard (60 AD)

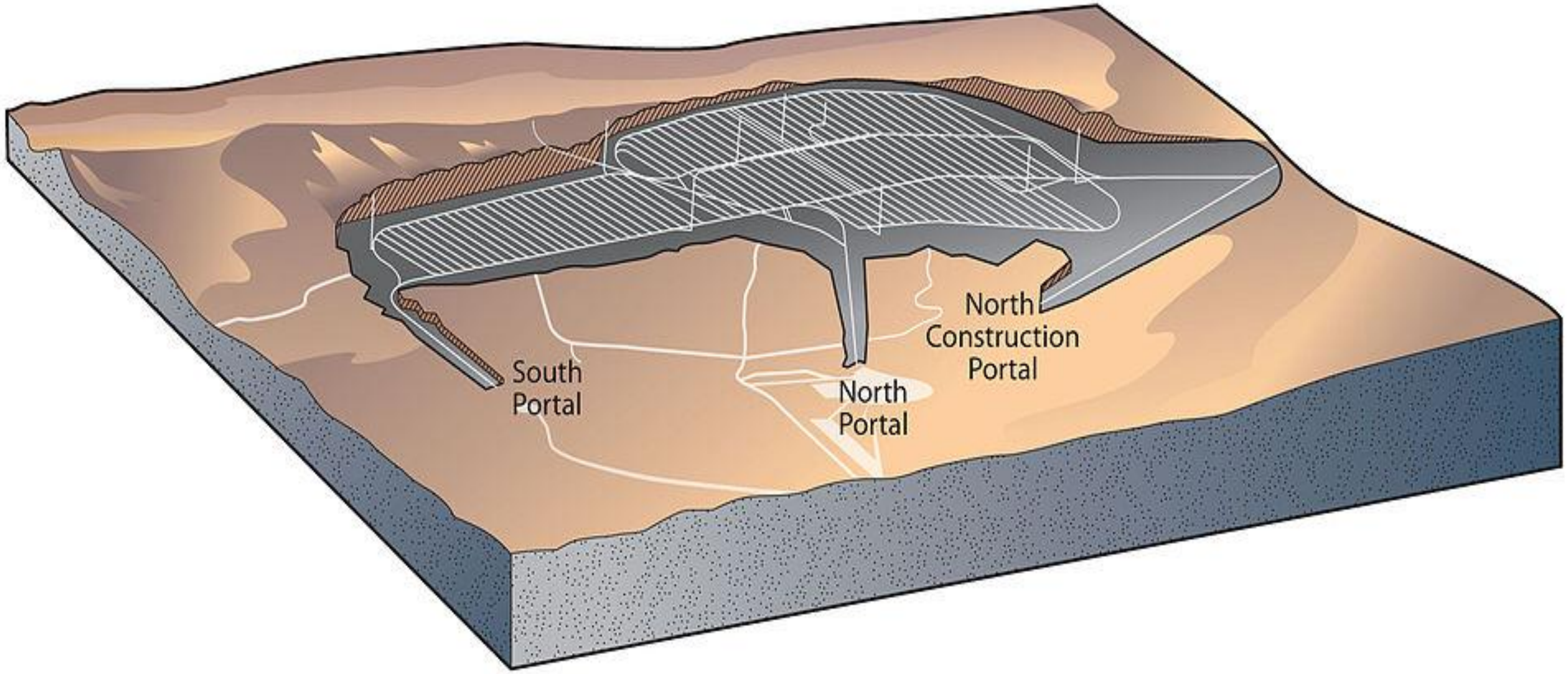
Future engineers may want to see our mathematics …

Gateway Arch (1965)

Save nuclear waste repository design data for many millennia …

[Mraz, Knowledge Preservation for Nuclear Waste Repositories, 2018, link]

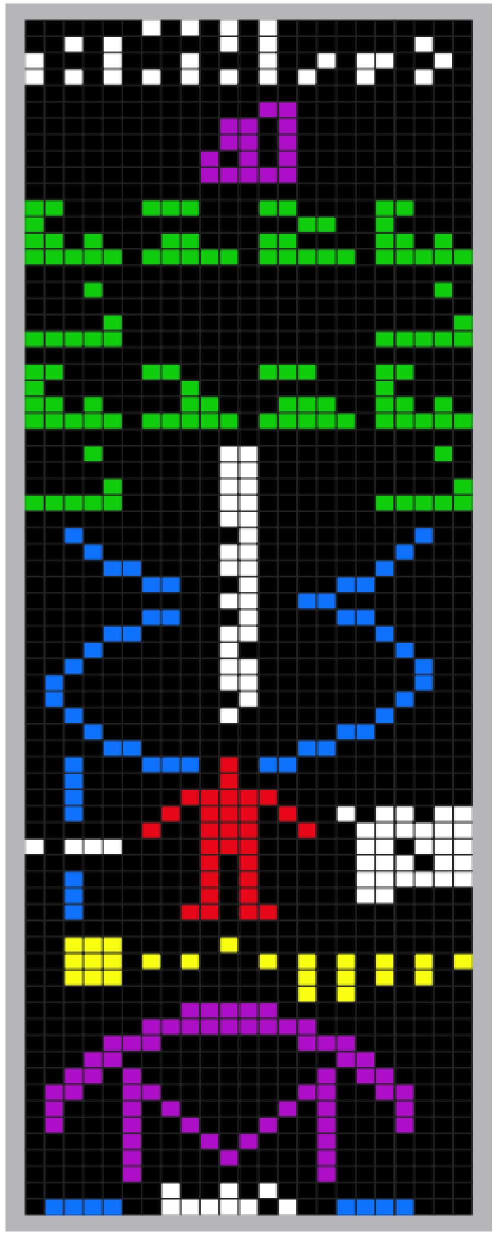
[Ryden, Extreme long-term preservation of information – who cares?, link]



Yucca Mountain Nuclear Waste Repository (?)

A Thought Experiment – The Alien Archeology Problem

- Humanity has gone extinct
- A million years later some advanced aliens visit earth
- If they find some binary CAD data, can they figure it out?
- Maybe if straightforward representation using basic concepts!



Arecibo Message (1974)

Long-Term Preservation of Binary Data

- Read-only DVDs – century or two
- M-Disk – 1,000 years
- Glass disks – millions/billions of years

  SciAm, Data Saved in Quartz Glass Might Last 300 Million Years, 2013

  https://www.5dmemorycrystal.com/
- DNA – million years

STEP
Standard for the Exchange of Product Model Data

- Enormous
- But no programmability
- May be good for a century

My Principles of Ultra-Long-Term CAD Preservation

http://axiomaticlanguage.org/LOTAR_Thoughts.html

1. Preserve operations and inputs, instead of results
2. Save “exact definitions” instead of approximations
   - Geodesic curves, symbolic constants (cos30deg, 0.1)
3. We need reproducibility of results
4. Save the geometric engine

Principles of Ultra-Long-Term CAD Preservation (2)

5. Need identical reproducibility – down to the last bit!
6. Floating point needs explicit definition
7. We need the geometric engine in source form
8. The programming language would be the standard

Principles of Ultra-Long-Term CAD Preservation (3)

9. You’d better pick a good programming language!

- Attributes:
  - Minimal, elegant
  - Founded on basic mathematical and computer science concepts
  - Extensible for representing geometry
- One candidate: a minimal Lisp
- My pick: “axiomatic language”

Axiomatic Language – Goals

1. pure specification – what, not how
2. minimal, but extensible
3. metalanguage – can define new language features

Axiomatic Language – Main Idea
Specification by Enumeration

- Program defined by set of symbolic expressions
- These enumerate inputs and corresponding outputs
- External behavior defined without algorithm
- Language defines these sets

The Core Language - Expressions

Axioms generate valid expressions.

expression:

  atom, – `` `abc ``

  expression variable, – `%x`

  or **sequence** of >=0 expressions and **string variable**s – `$1`

    – ``(`M () $a %)``

The Core Language – Axioms

axiom: **conclusion** expression and >=0 **condition** expressions

```
<conclu> < <cond1>, …, <condn>.
<conclu>.                ! a comment
```

The Core Language – Axiom Instances

Substitute values for variables

```
  (`A %x $1)    < (`B %x), (`C $1).
→ (`A `x `u ())< (`B `x), (`C `u ()).
```

The Core Language – Valid Expressions

If all conditions of an axiom instance are valid expressions, the conclusion is a valid expression.

```
(`a `b).
((%) $ $)< (% $).
```

```
→     (`a `b),
    ((`a) `b `b),
   (((`a)) `b `b `b `b),
          …
```

Syntax Extensions

```
'A'  =  (`char (`0 `1 `0 `0  `0 `0 `0 `1))

(… 'abc' …)  =  (… 'a' 'b' 'c' …)

"abc"  =  ('abc')  =  ('a' 'b' 'c')

abc  =  (` "abc")
```

Summary – A Vision for CAD (1)

http://www.axiomaticlanguage.org/A_Vision_for_CAD_released.pdf

1. Textual, human-readable engineering design language
   - Instead of CAD vendor's secret, proprietary, binary file format
   - High-level definitions instead of megabytes of numbers
   - Screen image, manufacturing, etc., generated from definitions
   - Programmability would support design automation & optimization

Summary – A Vision for CAD (2)

1. Textual, human-readable engineering design language
2. Open-source geometric engine
   - Accessible geometric algorithms – not a black box
   - Easier customization, easier migration
   - Archive geometric engine with design definitions

Summary – A Vision for CAD (3)

1. Textual, human-readable engineering design language
2. Open-source geometric engine
3. Implementation in axiomatic language
   - Powerful declarative language for specification
   - Good host for domain specific language for engineering design
   - Minimal & elegant – good long-term standard
   - Well-suited to proof

Challenge:

Automatically transform specifications to efficient programs!

[Baby Steps]