Recent advances in logical syntax of Universal IT Architecture (UITA)

Abstract: We introduce the ensemble named Universal IT Architecture (UITA) as based on software flow diagram symbols, DoD-STD-2167A/2168, and Meth8/VŁ4. We show the respective pairings of four symbols, tiers, and logical syntax, then to expand theoretical examples. We evaluate two practical examples of other systems: ISO 42010 and Archimate 3.1 as *not* tautologous. These results form a *non* tautologous fragment of the universal architectural logic UITA.

We assume the method and apparatus of Meth8/VŁ4 with Tautology as the designated proof value, **F** as contradiction, N as truthity (non-contingency), and C as falsity (contingency). The 16-valued truth table is row-major and horizontal, or repeating fragments of 128-tables, sometimes with table counts, for more variables. (See ersatz-systems.com.)

LET ~ Not, ¬ ; + Or, ∨, ∪, ⊔ ; - Not Or; & And, ∧, ∩, ⊓, ·, ∘ , ⊗ ; \ Not And;
> Imply, greater than, →, ⇒ , ↦, ≻, ⊃, ↠ ; < Not Imply, less than, ∈, ≺, ⊂, ⊬, ⊭, ↞, ≲ ;
= Equivalent, ≡, :=, ⇔, ↔, ≜, ≈, ≃ ; @ Not Equivalent, ≠, ⊕;
% possibility, for one or some, ∃, ∃!, ◊, M; # necessity, for every or all, ∀, □, L;
(z=z) T as tautology, ⊤, ordinal 3; (z@z) **F** as contradiction, Ø, Null, ⊥ , zero;
(%z>#z) N as non-contingency, Δ, ordinal 1; (%z<#z) C as contingency, ∇, ordinal 2;
~(y < x) (x ≤ y), (x ⊆ y), (x ⊑ y); (A=B) (A~B).
Note for clarity, we usually distribute quantifiers onto each designated variable.

1. Introduction

Systems to describe the logical syntax of IT architectures typically are top-down, from abstract, to particular, and lacking detail if unimplemented. The instant approach is bottom-up, commencing at software flow diagrams as commonly accepted symbols. The development scheme is to mirror at any stage or level the detail of DoD-STD-2167A/2168 rather than the minimal Mil-STD-498. The former life-cycle management is based on tailoring out concepts from a comprehensive pool of requirements, rather than the latter superficial outline based on adding concepts from a wish list of IEEE documents to minimize testing effort. The method adopted to enforce quality in meeting requirements is to falsify any part of the IT architecture using Meth8/VŁ4, the universal modal model logic checker. This gives finer detail in proof table results using the four-valued of contradiction, falsity, truthity, and tautology. The ensemble is named Universal IT Architecture (UITA).

2. Preliminary symbols

These are generalized to fit into four steps.

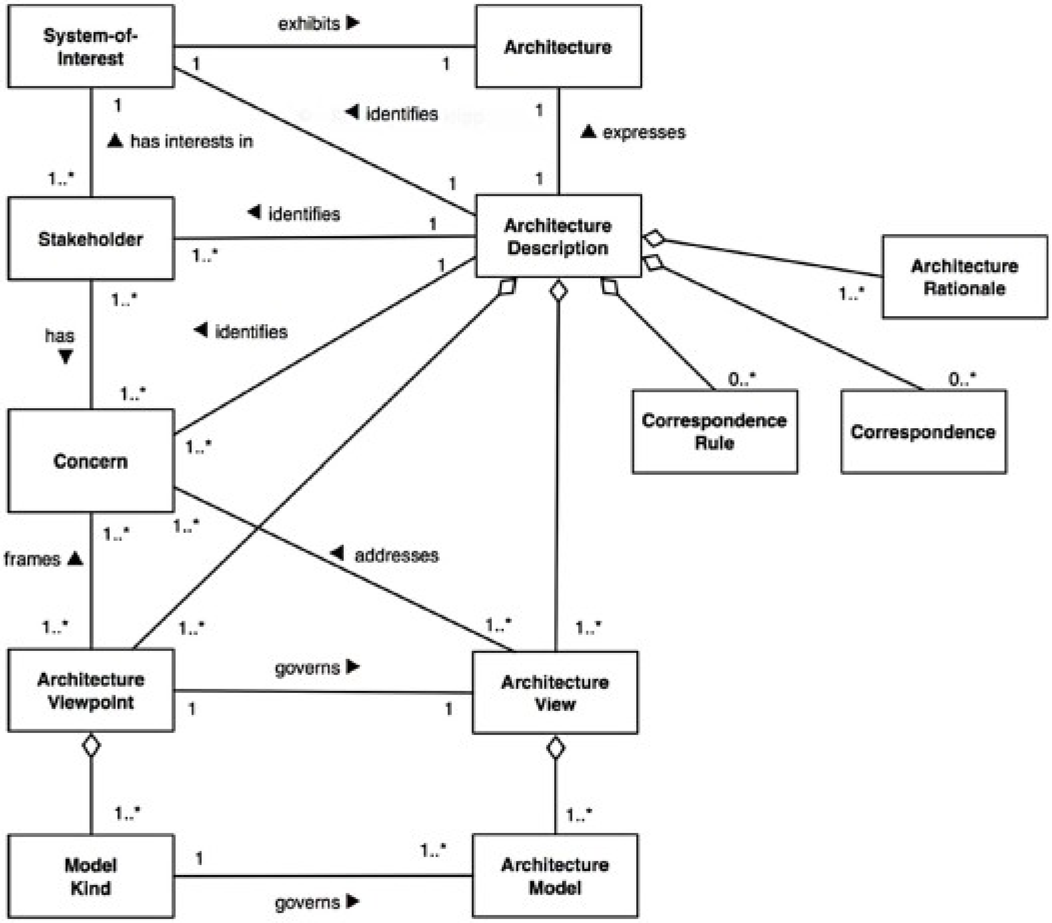
Eq.	*Software flow*		*IT tiers*		*Logical syntax*
1	Process	[] box	Presentation user layer	(PUL)	Propositional variable p,q,r,s,t,u,v,w,x,y,z
2	Decision	<> diamond	Application semantics layer	(ASL)	Operator; connective: ~; &, +, >, =
3	Input	() ellipse	Business logic layer	(BLL)	Antecedent, assume, possibility, for one %
4	Output	{ } page	Database persistent layer	(DPL)	Consequent, goal, necessity, for all #

3. Theoretical examples of logical syntax

Eq.	*Software flow*	*IT tiers*	*Logical syntax*
5	[p]——————[q]	[p] → [q]	p>q; q>p: `T`**`F`**`TT T`**`F`**`TT; TT`**`F`**`T TT`**`F`**`T`
6	[p]—— <r> ——[q]	[p] → <r> → [q]	%p>(r>#q); #p>(r>%q): `TTTT N`**`F`**`NN; TTTT TCTT`

4. Practical examples from other systems

From: International standards organization (ISO). (2011). ISO/IEC/IEEE 42010. Systems and software engineering architecture description.



(4.1.1)

Remark 4.1.1: We take architecture viewpoint and view as one item with model kind and architecture model supporting it.

LET	p	system of interest,
	q	architecture,
	r	stakeholder
	s	architecture description,
	t	architecture rationale,
	u	concern,
	v	correspondence rule,
	w	correspondence
	x	{architecture viewpoint,
		{architecture view,

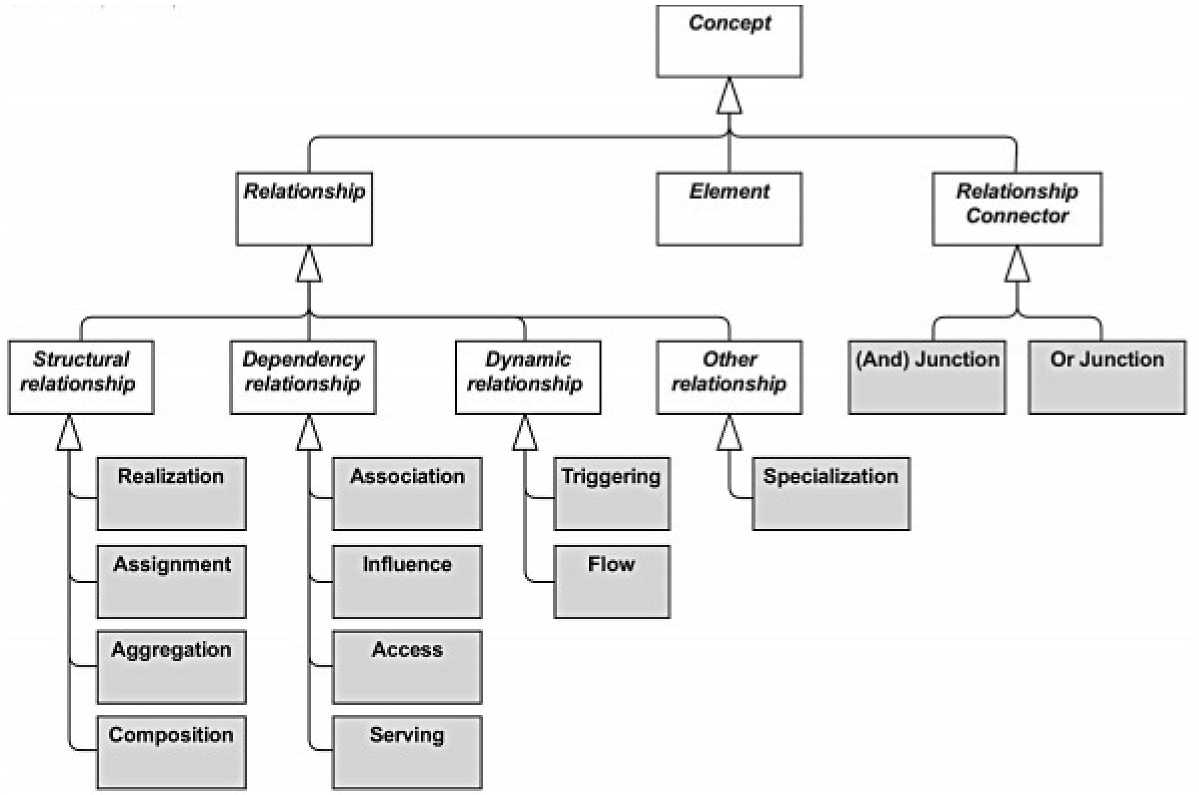
y model kind,
z architecture model.

(((((#x=s)>#u)&(#v&(#w&#t)))>%s)=(%p>(%q>%s)))&
((((#y&#z)=x)&(%y>#z))&(#r<(#u&(%s>%p)))) ;

```
FFFF NNNF FFFF NNNN( 2)}4
FFFF FFFF FFFF NFNF( 2)}
FFFF FFFF FFFF FFFF(48)
FFFF NNNF FFFF NNNN( 2)}4
FFFF FFFF FFFF NFNF( 2)}
FFFF FFFF FFFF FFFF(32)
FFFF NNNF FFFF NNNN( 2)}4
FFFF FFFF FFFF NFNF( 2)}        (4.1.2)
```

Remark 4.1.2: Eq. 4.1.2 as rendered is *not* tautologous, hence refuting the architecture description metamodel of the ISO/IEC/IEEE 42010 standard.

From: The open group. (2019). Archimate 3.1 specification.
pubs.opengroup.org/architecture/archimate3-doc/, 2019



(4.2.1)

Remark 4.2.1: The types under Relationship may be collapsed into itself as 2nd tier objects. Similarly the shaded objects may be collapsed into their respective 2nd tier objects without designation. This merge is convenient to keep the number of seven objects lesser than or equal to the 11 variables allowed in the standard edition of Meth8/VŁ4.

LET p concept,
q element,
r structural relationship,

s	dependency relationship,
t	dynamic relationship,
u	other relationship,
v	relationship connector.

```
(((q&r)&(s&t))&(u&v))>p ;  TTTT TTTT TTTT TTTT(7)}32
                           TTTT TTTT TTTT TTTF(1)}        (4.2.2)
```

Remark 4.2.2: Eq. 4.2.2 is *not* tautologous, hence refuting the Archimate concept meta model.

5. Discussion and conclusions

Two practical examples from other systems of ISO 42010 and Archimate 3.1 are refuted by the syntactical logic of Meth8/VŁ4 in ensemble system UITA. We anticipate mapping other IT architectures with results.