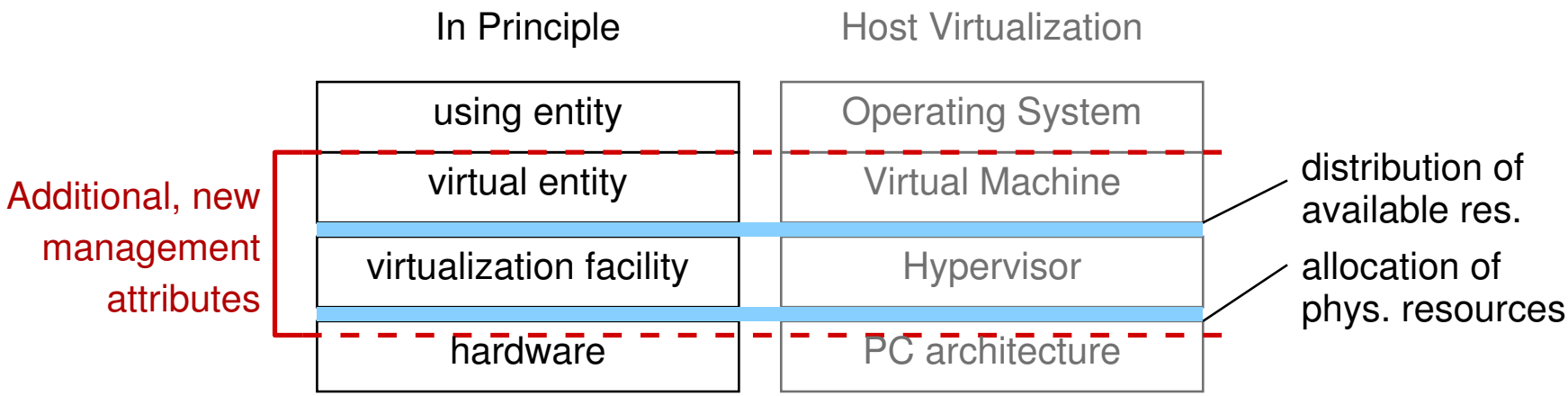
Bottom-up harmonisation of management attributes describing hypervisors and virtual machines

Vitalian A. Danciu, Nils gentschen Felde, Michael Kasch, Martin G. Metzker

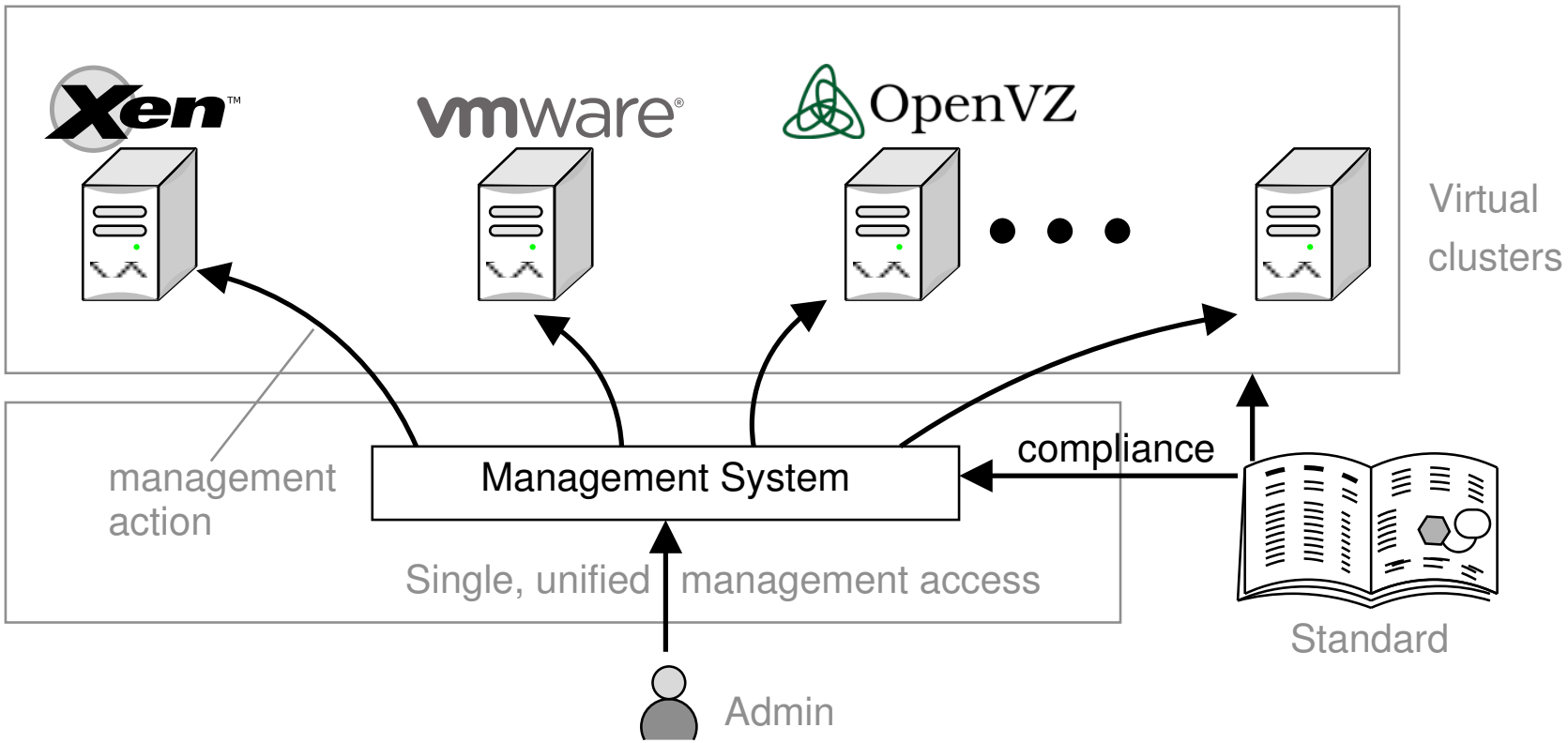
SVM 2011, Paris

New layers ⇒ new attributes



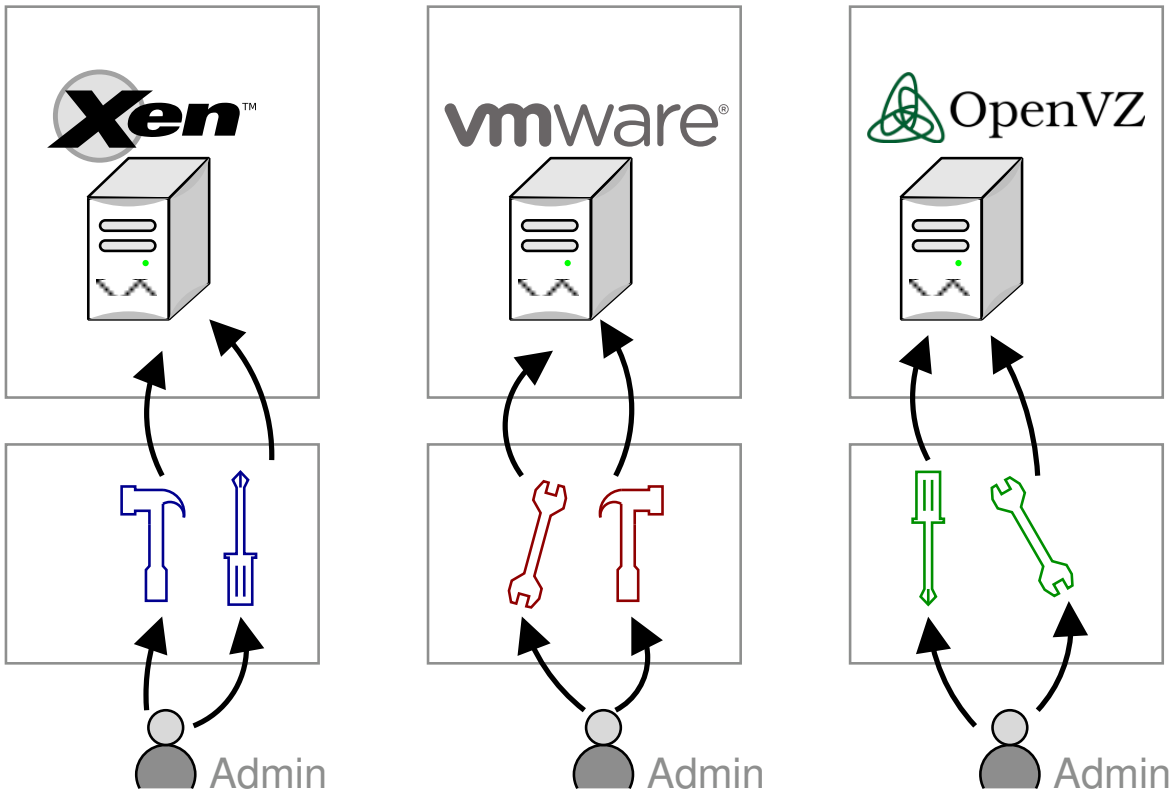
Larger number of attributes, representing different management views.

In an ideal world, far, far away…



Management information available as homogeneous, standards conform attributes: supports unified tool set.

Admin reality



Virtual clusters

Mgmt. Tools

Heterogeneous attribute set.

Tool sets necessarily different (may require different skill sets, as well).

Dimensions of difference

MNM TEAM

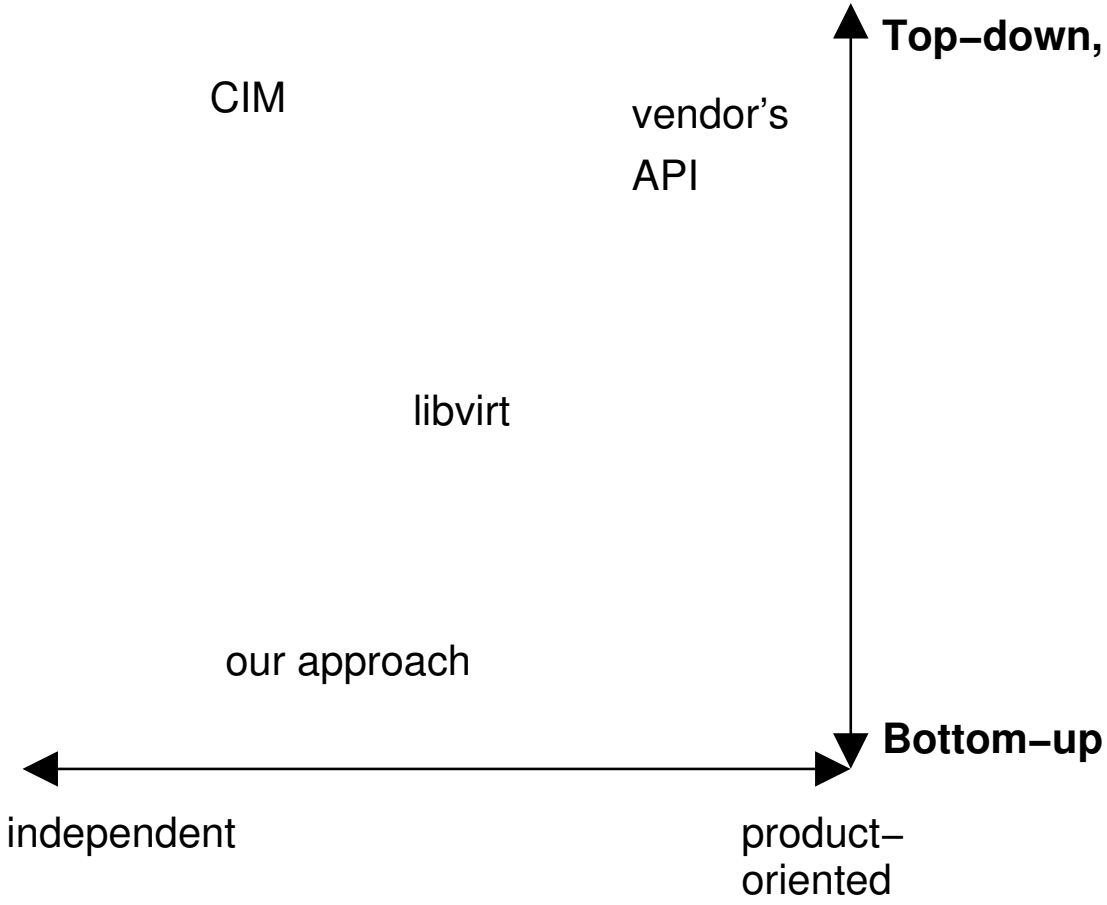
Syntax	name, structure, position, data type, precision
Semantics	meaning, value scale, unit, bounds
Access	protocol, function call, parameters, location, validity/rate-of-change

Example: Determine host memory use

	Name	Unit	Description
VMware	memorySize	byte	total host memory
	overallMemoryUsage	MByte	consumed host memory
Xen	memory_total	byte	total host memory
	memory_free	byte	free host memory
CIM	EndingAddress	KiB	ending address of highest memory
	BlockSize		size of a memory block
	ConsumableBlocks		available memory blocks

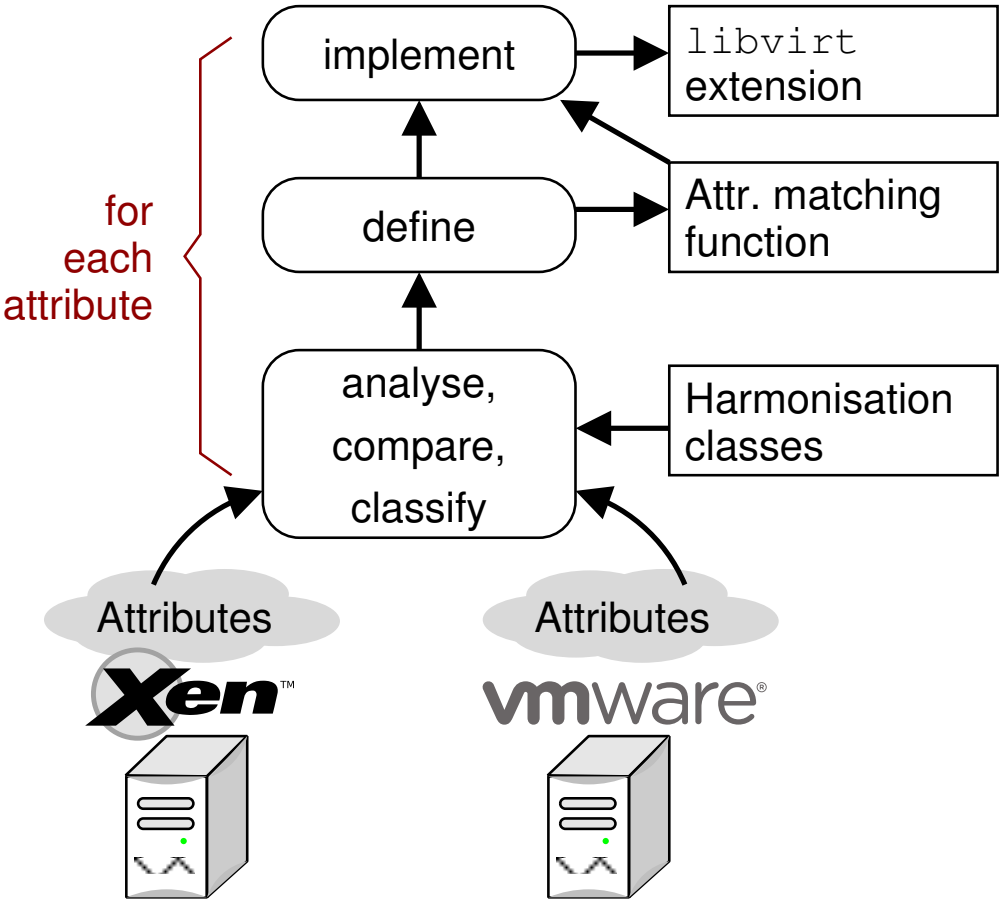
How to approach harmonisation?

Goal: Single representation for each attribute.

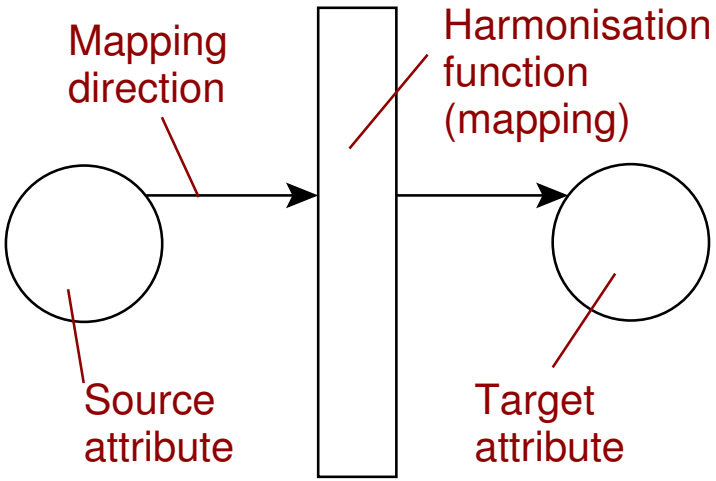


Bottom-up harmonisation

- Systematic processing of attributes
- Conceptual framework for harmonisation

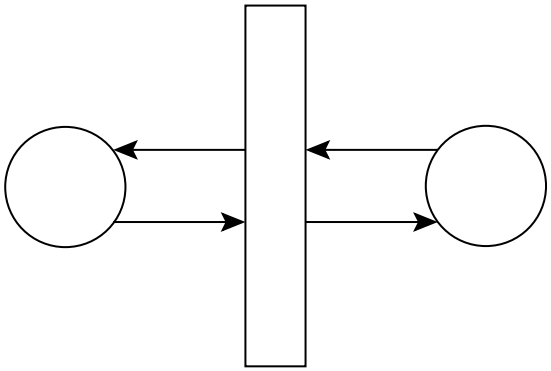


Classes of harmonisation functions

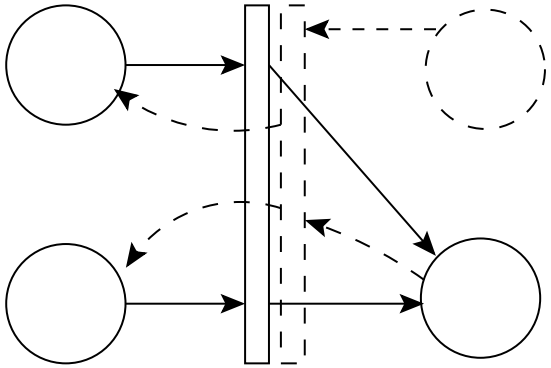


- Source/Target: hypervisor type, standard, . . .
- “Unmappable”: no corresponding target
- “Trivial”: identical source/target

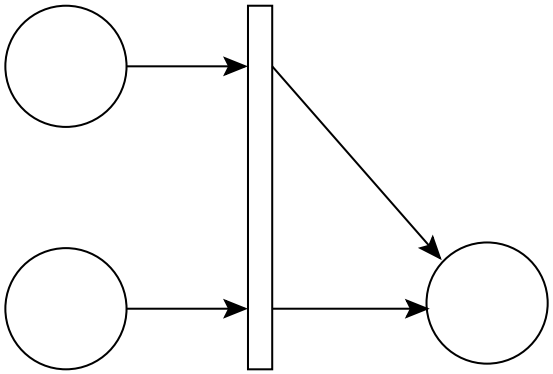
Bijective



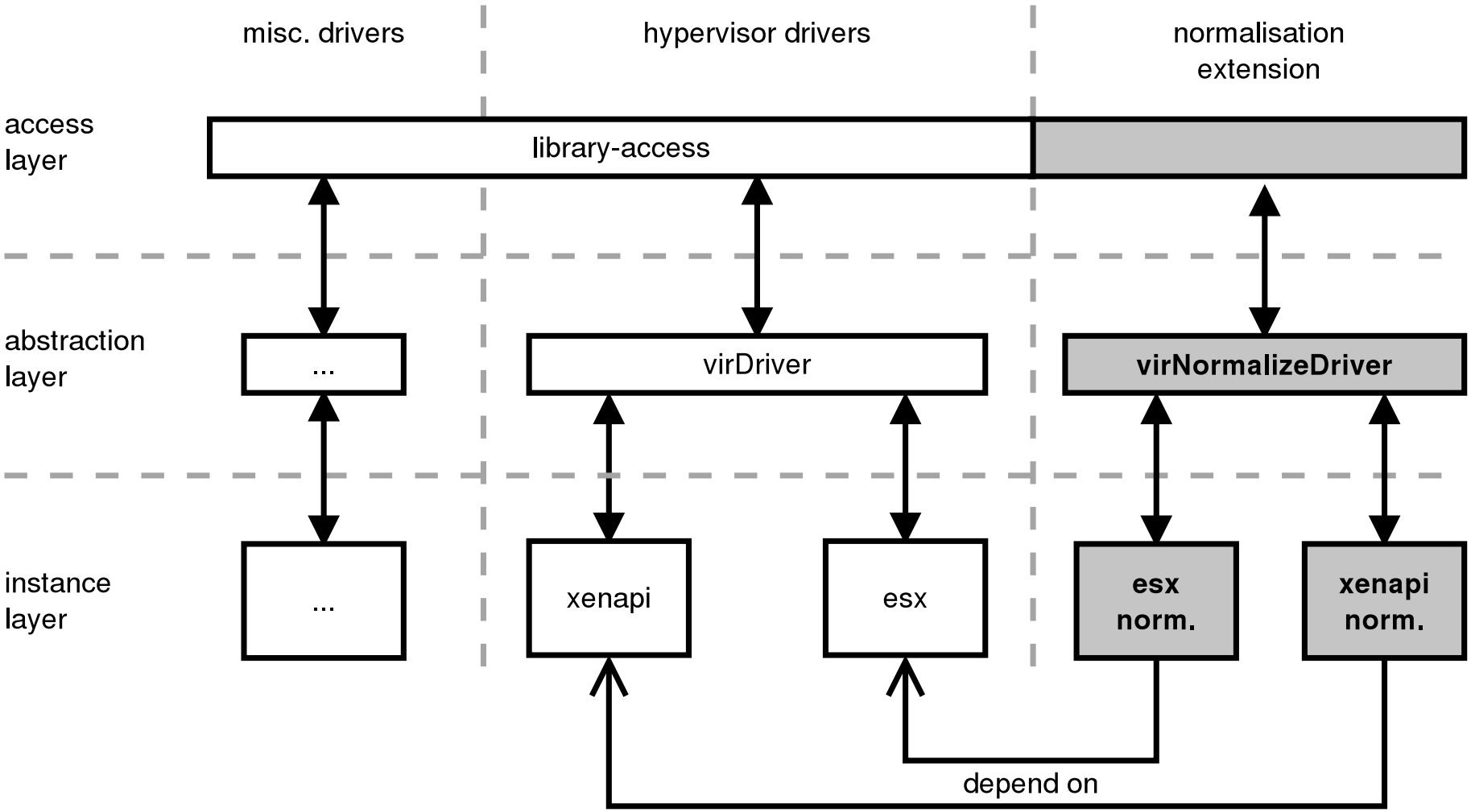
Reversible



Irreversible



Extension to `libvirt`



Discussion

Pros

- Harmonisation function classes guide analysis/implementation effort
- Reasonable effort to integrate attribute
- Extends established software

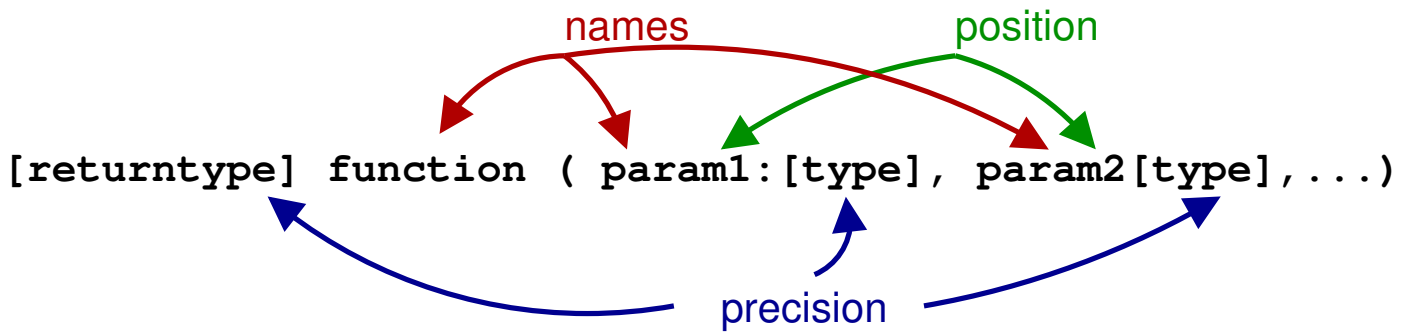
Cons

- Only attributes, so far; no explicit write a ccess
- Still needs semantic analysis
- Too tightly bound to implementation

Function invocation: can equivalence be established?

Need to invoke functions...

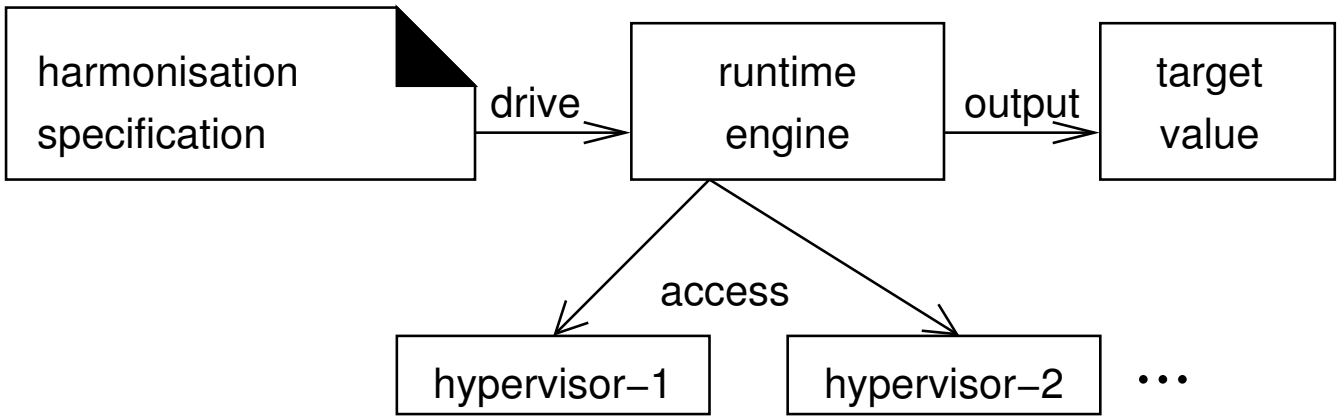
- by their respective name
- including parameters



- assumptions/pre-condition, effect/post-condition, action/side-effects
- call model, blocking/non-blocking, fault handling, ...

Idea: externalise harmonisation description

- Separate the specification for harmonisation from its implementation
  - Formal language to specify harmonisation functions
  - Interpreter of that language to encapsulate suitable access to MOs
- Enable collaborative effort
- Reuse of harmonisation specification (exchange)



Remember this!

- New, heterogeneous attributes due to virtualization.
  (differ in semantics, syntax, mode of access)
- This work: exploration of a bottom-up approach to harmonising attributes
  (function classes, extension to libvirt, read-only)
- Open issues: portable encoding, write access, harmonisation of management functions
- Possibility: external encoding of harmonisation function (facilitate exchange, decouple from implementation)

More questions?

Vitalian A. Danciu, Nils gentschen Felde, Michael Kasch, Martin G. Metzker
Munich Network Management Team, Ludwig-Maximilians-Universität München
<lastwordinname>@nm.ifi.lmu.de