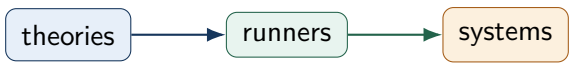
Sweedler Theory as an Architecture for Categorical Systems

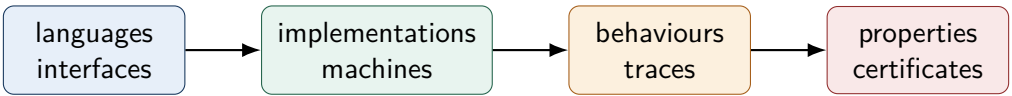
Runners, residual theories, and changing the ambient world

David Corfield



What problem is this trying to solve?

Categorical systems theory often has several partly separated layers.



Aim

Treat theory translation, stateful implementation, behaviour, and certification as parts of one compositional story.

Sweedler theory supplies the runner layer; changing the ambient category changes what systems can be.

Program

Around 300 pages of material **here**:

- Introduction
- Part I: Comodels
- Part II: Residual effects
- Part III: Tight direction
- Part IV: Duoidal structure
- Part V: Certification
- Part VI: Cost-enriched
- Part VII: Higher-dimensional execution
- Part VIII: Graph-based systems
- Part IX: Behaviours
- PolyInterlude: Polynomial functors
- Part X: Smooth systems
- Part XA: Probabilistic systems.
- Part XB: Analytic systems.
- Part XI: Traces
- Part XII: Internal language
- Conclusion
- Appendix A: IntRun via prisms
- Appendix B: Further good regulators
- Appendix C: Quantum runners
- Appendix D: Higher runners
- Arities: Arity-Generated Sweedler Universes

(NB: Early draft version)

Tight and loose directions of a double category, *Run*

Tight

A theory interpretation

$$f : T \longrightarrow T'$$

changes language without hidden state.

It says how each T-operation is expressed in T', preserving the equations.

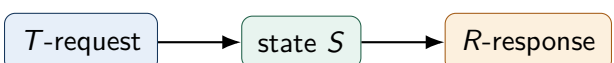
Loose

A runner

$$T \rightsquigarrow R$$

implements T-requests using residual R-behaviour and hidden state.

It is not just a translation of syntax; it is a machine.

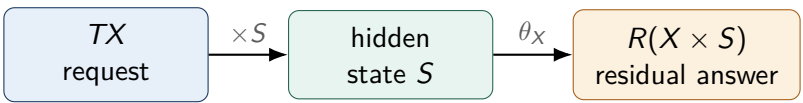


A concrete form of a runner

For accessible monads on sets, a runner can be remembered as a family of maps

$$\theta_X : TX \times S \longrightarrow R(X \times S),$$

compatible with unit and multiplication.



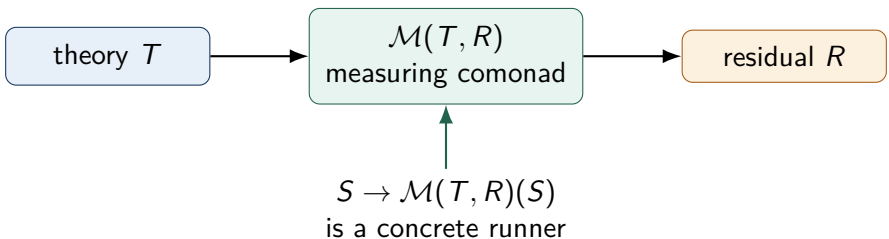
Reading

A T-program is executed by consulting and updating S, while emitting an R-effectful result.

Sweedler measuring: packaging all runners

The measuring comonad $\mathcal{M}(T, R)$ packages all ways of running T through R.

$$\{T, R\}\text{-coalgebras} \quad \simeq \quad \text{stateful runners } T \rightsquigarrow R.$$



Input data: a symmetric duoidal category

Take a symmetric duoidal category

$$(\mathcal{D}, \diamond, \star).$$

Theory side

Monoids for $\diamond$ are system theories:

$$T, R, P \in \mathsf{Mon}_{\diamond}(\mathcal{D}).$$

For endofunctors, $\diamond$ is composition and monoids are monads.

Runner side

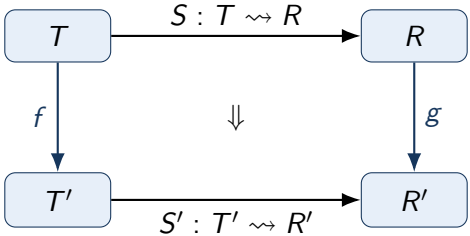
Sweedler homs measure one theory in another:

$$\mathcal{M}(T, R) \in \mathsf{Comon}_{\diamond}(\mathcal{D}).$$

Coalgebras are runners $T \rightsquigarrow R$.

The second monoidal product $\star$ is what makes measuring compositional.

The double category $\mathrm{Run}(\mathcal{D})$



- Objects: theories.
- Vertical arrows: tight theory interpretations.
- Horizontal arrows: loose stateful runners.
- Squares: compatibility of implementation with language change.

Why this matters

Ordinary semantics, effectful implementations, and interface refinement live in one diagrammatic environment.

The workhorse universe: $[\mathbf{Set}, \mathbf{Set}]_{\mathrm{acc}}$

Let

$$\mathcal{D}_{\mathbf{Set}} = [\mathbf{Set}, \mathbf{Set}]_{\mathrm{acc}}.$$

With composition $F \diamond G = F \circ G$, and Day convolution

$$F \star G = \mathrm{Lan}_{\times}(F \times G),$$

induced by the cartesian product of sets. $\diamond$-monoids are accessible monads.

Effects	Machines	Residuals
reader	Moore	identity
writer	Mealy	nondeterminism
exceptions	stack	probability
state/store	pop-push	cost/failure
	input/output	

A $\{T, R\}$-coalgebra is a machine accepting T-requests and answering by using R-effects.

Example 1: unary dynamics

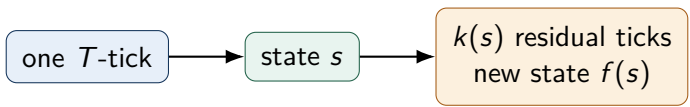
Let $T = R = \mathbb{N}y$, the free monad on one unary operation.

A runner is a stateful implementation of one step by residual steps:

$$\theta_X : TX \times S \longrightarrow T(X \times S).$$

Concretely, it has the form

$$(f, k), \qquad f : S \to S, \qquad k : S \to \mathbb{N}.$$



Already the runner is not merely a monad map: the cost and next state may vary with s.

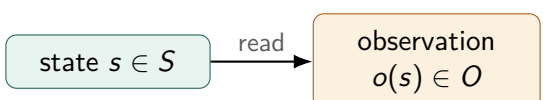
Example 2a: observations first

Begin with the theory which lets a computation ask for the current observation. For an observation set O, the reader theory is

$$T_O X = X^O.$$

A comodel of this theory is just an observable state space:

$$o : S \longrightarrow O.$$

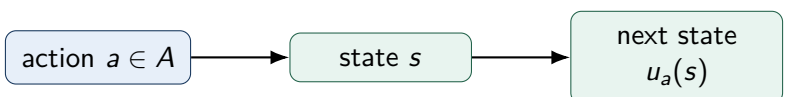


So the state is not yet acted on. It is only being exposed through an observation map.

Example 2b: then add actions

Now add commands $a \in A$ which change the state but do not themselves produce an observation. A comodel for the action part is an A-indexed transition system:

$$u : S \longrightarrow S^A, \qquad \text{equivalently} \qquad u_a : S \longrightarrow S \quad (a \in A).$$

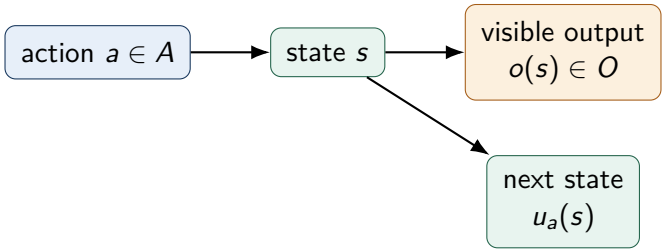


This is the controlled part: actions move the hidden state around.

Example 2c: putting observations and actions together

Combining the observation structure and the action structure gives the Moore-machine monad $T_{(A,O)-Moore}$ with comodels:

$$S \longrightarrow O \times S^A, \qquad s \longmapsto \big(o(s), a \mapsto u_a(s)\big).$$



The same Sweedler form then says how Moore-style observation can be implemented, translated, randomized, or certified.

Example 3: probability as residual theory

Let

$$\mathrm{rand}X = \sum_N \sum_{p \in \Delta_N} X^N$$

be the finite random-variable monad.

A runner

$$T \rightsquigarrow \mathrm{rand}$$

implements each T-operation by a randomized residual behaviour, such as a probabilistic Moore machine.

A rand-comodel

is a sampler, $\mathrm{rand} \rightsquigarrow Id$: a stateful source of choices resolving the residual random experiment.

Pulling it back

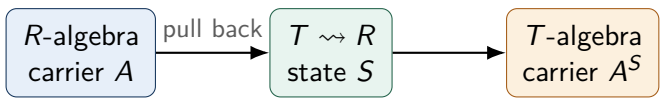
a $\{T, \mathrm{rand}\}$-runner plus such a sampler composes to an ordinary T-comodel, usually on enlarged state.

For example, a probabilistic Moore machine becomes an ordinary Moore machine once the random seed or sampling device is made part of the state.

Handlers: induction and delegation

A handler is the algebraic, evaluative side of the story: it says how to interpret the residual theory in some value object.

Given a runner with state S, handlers move along contravariantly.



There is also an adjoint transformation of T-handlers to R-handlers.

Point

Runners do not only simulate operations. They transport semantic interpretations of those operations.

Theories are not just interfaces

A signature specifies primitive operations and their arities — for example, an A-labelled action and an O-valued observation.

But a theory says more: it imposes equations between composites of those operations.

Signature

Which operations are available?
What are their input and output shapes?

Theory

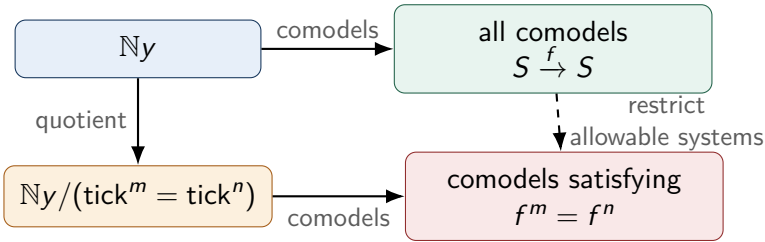
Which programs built from those operations are identified?
What laws must every model or comodel satisfy?

Quotienting the unary theory restricts comodels

Start with the free unary-operation theory:

$$\mathbb{N}y \qquad \text{one command tick} : X \to X.$$

A comodel is just an endofunction $f : S \to S$.



equivalently: a full subcategory of all endofunction systems

Point

The interface is still one button. The quotient theory says which one-button systems count: idempotent, periodic, or stabilising systems according to the equation imposed.

Composing and wiring theories

The category of theories has useful structure.

Construction	Systems reading
$T_1 + T_2$	superpose two effect vocabularies on shared state
$T_1 \amalg_S T_2$	glue theories along a shared interface/subtheory
coequalizer	impose identifications between interpretations
tensor $T_1 \otimes T_2$	impose commuting independent effects
other limits and colimits	...
parameterised theories	...

Message

We can build rich system theories from small ones before building particular systems.

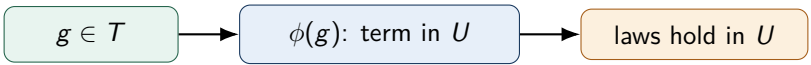
Theory interpretation after composition

A wiring diagram says how ports are connected. A theory interpretation says how one language is *programmed inside* another.

If T has generators g, a tight interpretation

$$\phi : T \longrightarrow U$$

sends each g to a U-term of the same arity, and respects the equations of T.



More than wiring

The target can be a composite theory:

$$U = U_{\text{obs}} + U_{\text{act}}, \qquad U = U_1 \otimes U_2,$$
$$U = (U_1 + U_2)/\text{protocol laws.}$$

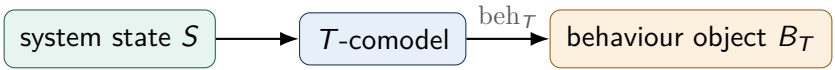
An external operation may therefore be realised by an internal protocol using observations, actions and equations.

From machines to behaviour

A T-comodel turns the abstract operations of T into concrete state transitions and observations.

For an accessible theory T on Set, there is a final T-comodel B_T, hence every T-system has a unique behaviour map

$$\mathrm{beh}_T : S \longrightarrow B_T.$$



B_T provides the objects of a category $\mathbb{B}_T$ which is such that $[B_T, \mathbf{Set}]$ is equivalent to all T-comodels.

(In *Poly*, there are categories $\mathbb{B}_{T,R}$. Runners in *Poly* form toposes.)

Reachability and observability

Sweedler theory separates two dual-looking questions.

Reachability

Starting from initial data, what states can be generated by the theory?

This is an initial or generated-subsystem problem.

Observability

Which states are indistinguishable by their final behaviour?

This is a final-comodel or behavioural quotient problem.

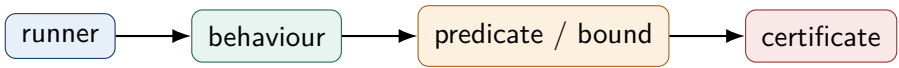
For linear systems this echoes Kalman duality; here it is a general categorical pattern.

Predicates, safety, and quantitative certification

Move from sets to valued predicates:

$$\mathcal{D} = [\mathbf{Pred}_V, \mathbf{Pred}_V]_{\mathrm{acc}}.$$

Now states carry properties, bounds, costs, or truth values from a quantale V.



Examples

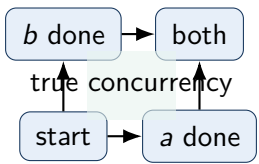
Safety invariants, assume–guarantee reasoning, liveness fragments, amortised cost, potential functions.

Concurrency: precubical ambient categories

Take

$$\mathcal{D} = [\mathbf{pCub}, \mathbf{pCub}]_{\mathrm{acc}}.$$

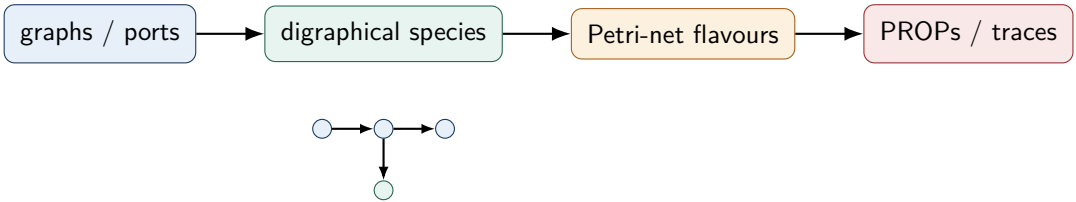
Then states and behaviours may be higher-dimensional, not merely sets of states with transitions.



A Sweedler calculus of concurrency should compare interleaving, independence, deadlock, and amortised resource use inside the same runner language.

Graph-based systems: species and networks

We can change the shape of systems rather than just adding state or probability.



Point

A theory can specify local graph operations; a runner implements them in a structured network of places, ports, transitions, guards, jumps, inhibitors or probabilities.

The ambient category can be a graph/species presheaf world, for example a coloured Petri-net setting such as

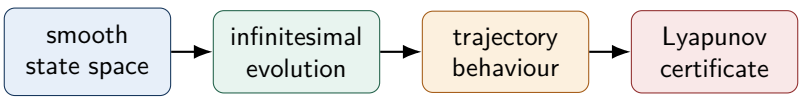
$$\mathcal{D}_S = [\mathbb{B}_S^{op} \times \mathbb{B}_S, \mathbf{Set}].$$

Smooth dynamics and infinitesimal change

In a smooth or synthetic-differential ambient category, the tangent functor can be represented internally:

$$TX = X^D$$

for an infinitesimal object D.



This is where runners can mediate between operational theories, differential structure, and control-theoretic certification.

A differential direction

The smooth story can be pushed further.

For a smooth state space X, the tangent bundle gives an internal category

$$TX \rightrightarrows X,$$

and its polynomial comonoid c_X has linear dual

$$[c_X, y] \cong \sum_{v \in \mathfrak{X}(X)} y^X.$$

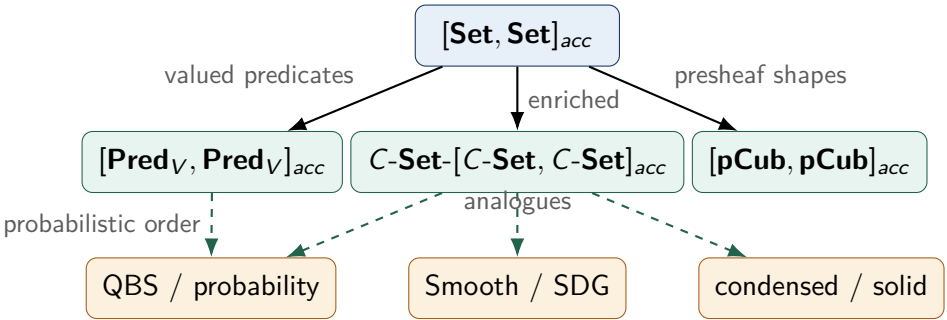
The operations are literally vector fields

Its Sweedler substitution is addition after state-dependent evaluation. Passing instead to the infinitesimal path groupoid brings in composition and the Lie bracket.

In the Cahiers topos, the jet comonad J^∞ takes the next step: its coalgebras are formally integrable PDEs.

vector fields $\longrightarrow$ infinitesimal dynamics $\longrightarrow$ jets/PDEs

Changing the ambient world



Design principle

The goal is not one final universe, but a functorially related family of $\mathrm{Run}(\mathcal{D})$'s.

A larger programme

The talk has been an introduction to a broad experiment.

- Introduction
- Part I: Comodels
- Part II: Residual effects
- Part III: Tight direction
- Part IV: Duoidal structure
- Part V: Certification
- Part VI: Cost-enriched
- Part VII: Higher-dimensional execution
- Part VIII: Graph-based systems
- Part IX: Behaviours
- PolyInterlude: Polynomial functors
- Part X: Smooth systems
- Part XA: Probabilistic systems.
- Part XB: Analytic systems.
- Part XI: Traces
- Part XII: Internal language
- Conclusion
- Appendix A: IntRun via prisms
- Appendix B: Further good regulators
- Appendix C: Quantum runners
- Appendix D: Higher runners
- Arities: Arity-Generated Sweedler Universes

Comparison with DOTS and interfaces

DOTS-style approaches put typed interfaces and wiring at the front.

Sweedler theory instead starts from theories and runners, but it does not give up wiring:

$$\text{tight interpretations} + \text{loose runners} + \text{theory constructions}$$

provide a large amount of interface and feedback structure internally.

Likely synthesis

Use DOTS-like typed interfaces as a base, and Sweedler measuring/runners in the fibres.

This leads naturally to $\mathsf{IntRun}(\mathcal{D})$: interfaces as pairs of theories, with prism-like loose morphisms.

Takeaways

1. Theories are not just interfaces: equations matter.
2. Sweedler homs make implementation of one theory in another into structure.
3. Coalgebras of measuring comonads are stateful runners.
4. Tight interpretations and loose implementations assemble into $\mathrm{Run}(\mathcal{D})$.
5. Residual theories encode probability, nondeterminism, cost, failure, and observation.
6. Changing $\mathcal{D}$ changes the ontology of systems: predicates, concurrency, smoothness, probability, analysis.

Thesis

Sweedler theory is a candidate architecture for categorical systems theory because it organizes languages, implementations, behaviours, and semantic change in one compositional framework.