Compositional Cyber-Physical Systems Modeling

Georgios Bakirtzis
University of Virginia
Charlottesville, VA USA
bakirtzis@virginia.edu

Christina Vasilakopoulou
University of Patras
Patras, Greece
cvasilak@math.upatras.gr

Cody H. Fleming
University of Virginia
Charlottesville, VA USA
fleming@virginia.edu

Assuring the correct behavior of cyber-physical systems requires significant modeling effort, particularly during early stages of the engineering and design process when a system is not yet available for testing or verification of proper behavior. A primary motivation for ‘getting things right’ in these early design stages is that altering the design is significantly less costly and more effective than when hardware and software have already been developed. Engineering cyber-physical systems requires the construction of several different types of models, each representing a different view, which include stakeholder requirements, system behavior, and the system architecture. Furthermore, each of these models can be represented at different levels of abstraction. Formal reasoning has improved the precision and expanded the available types of analysis in assuring correctness of requirements, behaviors, and architectures. However, each is usually modeled in distinct formalisms and corresponding tools. Currently, this disparity means that a system designer must manually check that the different models are in agreement. Manually editing and checking models is error prone, time consuming, and sensitive to any changes in the design of the models themselves. Wiring diagrams and related theory provide a means for formally organizing these different but related modeling views, resulting in a compositional modeling language for cyber-physical systems. Such a categorical language can make concrete the relationship between different model views, thereby managing complexity, allowing hierarchical decomposition of system models, and formally proving consistency between models.

1 Introduction

Cyber-physical systems are composed of computing platforms, control systems, sensors, and communication networks [20]. These systems provide critical service capabilities in a number of engineering domains, including transportation, medical devices, and power, to name a few. The design of cyber-physical systems poses new challenges because of the intertwined nature of digital control with physical processes and the environment. As autonomy and coordination between a multitude of such systems becomes commonplace, there is an increasing need to formally assure not only their individual behavior but also the emergent properties that arise from the behavior of the composite system. One such emergent property is safety. When cyber-physical systems exhibit unwanted behaviors, they can transition to hazardous states and then lead to accidents. To avoid such undesirable outcomes it is necessary to provide evidence of correct behavior before deployment, during the design phase of the systems lifecycle. Design changes in later stages of the system lifecycle cost more and are less effective [23]. It is only possible to produce this evidence early – when an implemented system is not yet available – through the management and use of various models.

The formal assurance of such complex systems requires the use of compositional methods to manage the various views necessary for the modeling and simulation of cyber-physical system design artifacts [8]. At the highest level of abstraction a system must be constrained by requirements, which define what the system ought to and ought not to do. Through an iterative process, requirements are refined to dictate the set of allowable behaviors and then used to architect a candidate implementation. There are often several

To appear in EPTCS.

candidate implementations that are functionally equivalent but implemented with different hardware or software and they can be compared in terms of how they respond to different performance metrics.

There is a gap in systems engineering, including its currently narrowly overlapping fields of safety and security, where there is a lack of semantic strength[1] between requirements, behaviors, and architectures. These are considered connected in practice but currently reside in distinct formalisms, leading to an unmanageable problem of complexity [12, 13]. However, each is naturally modeled in different formalisms, for example, first-order logic for requirements, difference and differential equations for physical behavior, and graphs for architectures. Indeed, state space representations in control are informally related to finite state machines in the software implementation. Compositional formal methods assist with making this informal understanding of relation and unification explicit.

Recent work in applied category theory for dynamical systems [10,24,26], probabilistic failure analysis [5], control [4], and requirements management [16] is particularly relevant to making this unification happen. Specifically, the wiring diagram category [27] is useful for compositionally modeling cyber-physical systems because it models reactive systems, feedback, and real-time computation, all of which are necessary properties for modeling cyber-physical systems [1, §1.2]. However, to date there is little work in merging this mathematical machinery for the design of a modeling language blueprint for cyber-physical systems using categorical, or otherwise algebraic, concepts.

Our aim with this line of research is to use categorical formalisms and concepts to better model systems for the purposes of system assurance. To achieve this aim we bring theoretical concepts from applied category theory to the realm of application by modeling and simulating cyber-physical systems in a categorical context. This requires us to precisely unify a number of diverse models for the purpose of having a holistic picture of the systems behavior and how it relates to candidate implementations.

We provide evidence that using category theory as a modeling language makes it possible to unify the necessary but diverse views of cyber-physical system models. Using category theory solves an important problem in systems theory *and* systems engineering practice, which is a lack of formal traceability between requirements, behaviors, and implementations. We posit that wiring diagrams provide an effective solution to formal unification of system models that will improve scalability of the overall modeling effort. As a first step, it is required that we reverse engineer the composite behavior of a system categorically, and combine these possible decompositions with appropriate hierarchical implementations to provide flexibility for the system designer: this is the main contribution of this paper. The results of this ongoing project, which uses real-life systems and design needs, exhibit the usefulness of category theory for finding consistency errors in system models early in the design phase. Applying formal tools and constructions directly to the field of systems engineering results in highly unified engineering design artifacts, which in turn have the potential to reduce the amount of faulty and dangerous systems that get deployed.

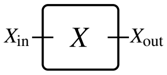
2 Categorical Semantics for Cyber-Physical Systems

In engineering there are two main iterative design steps, *synthesis* and *analysis*. Synthesis is the process by which new systems are formed from already known device laws, and analysis is the process by which the new system is evaluated as a whole for its overall function. At each iteration of synthesis and analysis there is a notion of *refinement*. Designing systems starts at some level of abstraction and as we progress through the design-phase the modeling artifacts are developed to be closer and closer to the eventual real system that will be deployed.

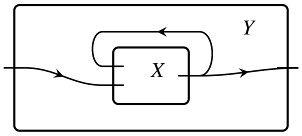
[1] The complement of semantic strength is a perhaps familiar term: semantic gap.

In this work, we follow the categorical formalism of wiring diagrams [24,27]. After we introduce the relevant theory in this section, we proceed to investigating a specific example of a cyber-physical system, namely an unmanned aerial vehicle (UAV). We start with modeling the system functions (Section 3) and iteratively decompose down to an implementation (Section 4) of an actual flight controller and airframe. We show that by using wiring diagrams we can synthesize system models, analyze them with respect to their behavioral outcomes, and refine them from a particular set of behaviors to a concrete candidate architecture.

Recall that for a category **C** with products, the category $\mathbf{W_C}$ of *labeled boxes* and *wiring diagrams*[2] has as objects pairs of objects $X = (X_{\text{in}}, X_{\text{out}})$ that should be thought of as the input and output types of an interface

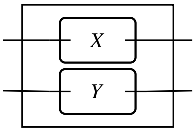


that serves as a placeholder for some system, whereas a morphism $X \to Y$ is a pair of arrows $(f_{\text{in}} \colon X_{\text{out}} \times Y_{\text{in}} \to X_{\text{in}}, f_{\text{out}} \colon X_{\text{out}} \to Y_{\text{out}})$ in **C** that should be thought of as providing the flow of information in a picture as follows



(1)

It is a monoidal category, with $X \otimes Y = (X_{\text{in}} \times Y_{\text{in}}, X_{\text{out}} \times Y_{\text{out}})$ expressing parallel placement



(2)

In what follows, **C** will be **Set** or **Lin** (the category of linear spaces and linear maps) whereas in general the types could be objects in any cartesian monoidal category, which provides the flexibility to model different sorts of information or views of systems as well as incorporate notions of discrete and continuous time or functorially map between systems with different values. We sometimes write just **W** for $\mathbf{W_C}$.

Morphisms in this category give a concrete mathematical description of pictures like Figure 1 where the interconnection of three systems $\mathtt{L}, \mathtt{C}, \mathtt{D}$ produces a composite system $\mathtt{U}$. As objects in $\mathbf{W_{Set}}$, these are $\mathtt{L} = \mathtt{C} = \mathtt{U} = (\mathbb{R}^2, \mathbb{R})$ and $\mathtt{D} = (\mathbb{R}, \mathbb{R})$, whereas the morphism $\mathtt{L} \otimes \mathtt{C} \otimes \mathtt{D} \to \mathtt{U}$ that captures the above arrangement is explicitly given in Figure 2.

The essence of the proposed categorical formalism for systems modeling is encapsulated by viewing different types of system behaviors or requirements as *algebras* for **W**, namely lax monoidal functors[3] from **W** to $(\mathbf{Cat}, \times, \mathbf{1})$. So each algebra assigns to a box $X = (X_{\text{in}}, X_{\text{out}})$ a category FX of systems that can be placed inside the box, and also assigns to a wiring diagram $(f_{\text{in}}, f_{\text{out}}) \colon X \to Y$ a functor $FX \to FY$ that, given a system inhabiting the internal box, produces the *composite* system inhabiting the external box. Moreover, the laxator $\phi_{X,Y} \colon FX \times FY \to F(X \otimes Y)$ maps two given systems inside the boxes in wiring diagram (2) to a system inside their parallel placement.

[2] The original definition involves **C**-*typed finite sets* and functions that respect the types; herein we abuse the notation by referring to the resulting category by taking products of types [24, Rem. 2.2.2].

[3] These are in fact usually pseudofunctors and more specifically *monoidal indexed categories* [24, Rem. 2.1.1].

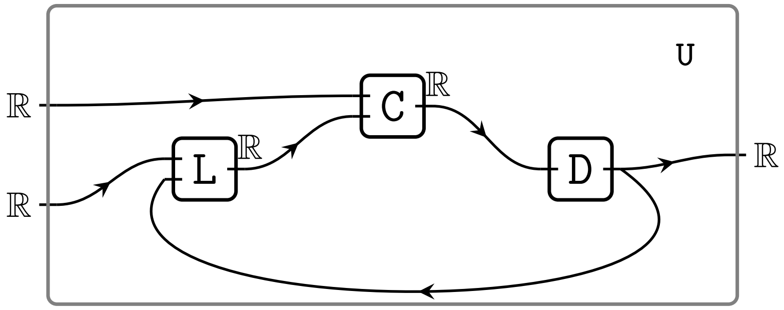
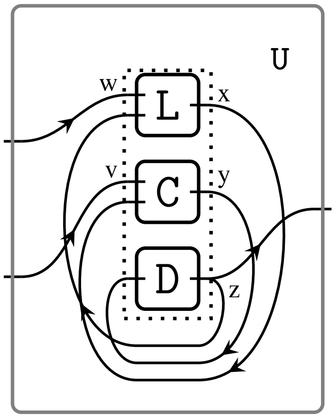


Figure 1: Wiring diagrams make explicit the type of interconnections between subsystems.



$$f_{\text{in}}: \overbrace{\mathbb{R}^3}^{(L\otimes C\otimes D)_{\text{out}}} \times \overbrace{\mathbb{R}^2}^{U_{\text{in}}} \to \overbrace{\mathbb{R}^5}^{(L\otimes C\otimes D)_{\text{in}}} \quad \text{by } (x,y,z,w,v)\mapsto(w,z,v,x,y)$$

$$f_{\text{out}}: \underbrace{\mathbb{R}^3}_{(L\otimes C\otimes D)_{\text{out}}} \to \underbrace{\mathbb{R}}_{U_{\text{out}}} \quad \text{by } (x,y,z)\mapsto z$$

Figure 2: In wiring diagrams the syntax defines the architecture; that is, how the boxes are arranged.

One of the examples used in this work is the algebra of *linear time-invariant systems*, also called *linear dynamical systems* by Spivak [25]. The algebra, namely a lax monoidal functor

$$\text{LTIS}\colon \mathbf{W_{Lin}} \to \mathbf{Cat} \tag{3}$$

assigns to any box $X_{\text{in}}\,\boxed{}\,X_{\text{out}}$ a special discrete dynamical system; that is, a set S called the *state space* and two functions u, r called *update* and *readout* as in

$$(S, u\colon S\times X_{\text{in}} \to S, r\colon S \to X_{\text{out}})$$

where all $S, X_{\text{in}}, X_{\text{out}}$ are linear spaces and both update and readout functions u and r are linear functions expressed as

$$u(s,x) = \mathscr{A}\cdot s + \mathscr{B}\cdot x = \begin{pmatrix}\mathscr{A} & \mathscr{B}\end{pmatrix}\begin{pmatrix}s\\ x\end{pmatrix} \qquad r(s) = \mathscr{C}\cdot s$$

where $\mathscr{A}$, $\mathscr{B}$ and $\mathscr{C}$ are matrices of appropriate dimension and $\cdot$ is matrix multiplication. For example, if $X_{\text{in}} = \mathbb{R}^k$, $X_{\text{out}} = \mathbb{R}^\ell$ and $S = \mathbb{R}^n$, then $\mathscr{A} \in {}_nM_n$ represents a transformation $\mathbb{R}^n \to \mathbb{R}^n$, $\mathscr{B} \in {}_nM_k$ a transformation $\mathbb{R}^k \to \mathbb{R}^n$ and $\mathscr{C} \in {}_\ell M_n$ a transformation $\mathbb{R}^n \to \mathbb{R}^\ell$. We often write $\overset{\bullet}{s} = u(s,x)$ as a standard notation, even in this discrete time-case.

Now given an arbitrary wiring diagram $(f_{\text{in}}, f_{\text{out}})\colon (X_{\text{in}}, X_{\text{out}}) \to (Y_{\text{in}}, Y_{\text{out}})$, notice that for $Y_{\text{in}} = \mathbb{R}^{k'}$ and $Y_{\text{out}} = \mathbb{R}^{\ell'}$ both components of the wiring diagram are also expressed via matrices

$$F_{\text{in}} = \begin{pmatrix}{}_k(\mathscr{A}^F)_\ell & {}_k(\mathscr{B}^F)_{k'}\end{pmatrix} \qquad F_{\text{out}} = {}_{\ell'}\mathscr{C}^F_\ell$$

The functor LTIS maps some system $(S, \mathscr{A}, \mathscr{B}, \mathscr{C})$ in $\mathbb{R}^{k} \,-\!\boxed{X}\!-\, \mathbb{R}^{\ell}$ to the composite system

$$(S, \mathscr{A} + \mathscr{B} \cdot \mathscr{A}^{F} \cdot \mathscr{C},\ \mathscr{B} \cdot \mathscr{B}^{F},\ \mathscr{C}^{F} \cdot \mathscr{C}) \tag{4}$$

in $\mathbb{R}^{k'} \,-\!\boxed{Y}\!-\, \mathbb{R}^{\ell'}$. The laxator of LTIS maps any two such systems $(S_1, \mathscr{A}_1, \mathscr{B}_1, \mathscr{C}_1)$, $(S_2, \mathscr{A}_2, \mathscr{B}_2, \mathscr{C}_2)$ inhabiting parallel boxes to the system

$$\left(S_1 \times S_2, \begin{pmatrix} \mathscr{A}_1 & 0 \\ 0 & \mathscr{A}_2 \end{pmatrix}, \begin{pmatrix} \mathscr{B}_1 & 0 \\ 0 & \mathscr{B}_2 \end{pmatrix}, \begin{pmatrix} \mathscr{C}_1 & 0 \\ 0 & \mathscr{C}_2 \end{pmatrix}\right).$$

This is the machinery that will allow us to model control systems and, therefore, the largest class of cyber-physical systems.

3 Simulating Dynamics

A UAV is often a small factor fixed wing plane or multi-rotor copter that is controlled on the ground by a computer. The unmanned aerial system, which includes the UAV, ground control station, and a network between the two, is responsible for navigating the plane from one waypoint to the next.

A UAV includes a control system and as such it needs to adhere to certain control laws, which have to be designed, implemented, and tuned. But what is a *behavioral* interpretation of what a UAV does? We can consider behavior as a composition of system functions. In this sense, a compositional interpretation could be the composition of its sensors, controllers, and control surfaces to produce a known set of dynamics. To simulate the UAV behavior the first step is to produce the dynamic response of the system to environmental values. We capture the dynamics of such a system to the systems-as-algebras paradigm, where we consider that the sensors are measuring the environment but also get input from the resulting airframe dynamics. We describe this process step-by-step in a practical approach, and conclude the section with the bare categorical dimension of said process.

Thinking of the composite UAV system in terms of its control laws for discrete time, we would model its dynamics by its difference equations – since we are operating in discrete time – but we map the matrix representation as above. Therefore

$$s_{p+1} = \mathscr{A} s_p + \mathscr{B} c_p$$

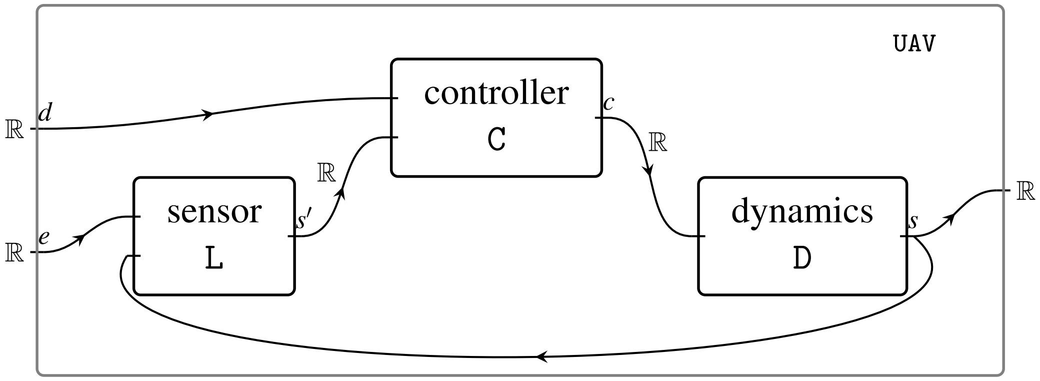
where $s_p \in \mathbb{R}^n$ is the discrete time state and $c_p \in \mathbb{R}^n$ is the control signal/output, and

$$y_p = \mathscr{C} s_p + \mathscr{D} c_p$$

is the measurement which is also $y_p \in \mathbb{R}^n$, where we could assume a perfect measurement so that $s_p = y_p$. In practice, the airframe behavior (say, the static parts of the airframe) can be considered or otherwise contained in $\mathscr{A}$, servos and actuators would be part of $\mathscr{B}$, sensors are part of $\mathscr{C}$. The disturbance, or feedforward term, $\mathscr{D}$ is typically 0 of proper dimension. We define the following notational abbreviations:

$$\begin{aligned} \mathtt{L} &\triangleq \text{ sensor,} \\ \mathtt{C} &\triangleq \text{ controller, and} \\ \mathtt{D} &\triangleq \text{ dynamics.} \end{aligned}$$

Any such system requires a prediction s', this is typically used within $\mathtt{C}$ (Figure 3). This is done in different ways (open-loop, MPC, classical control, et cetera). In modern control, one typically takes a



$$f_{\text{in}} \colon \mathbb{R}^3 \times \mathbb{R}^2 \to \mathbb{R}^5,\ (s', c, s, e, d) \mapsto (e, s, d, s', c)$$
$$f_{\text{out}} \colon \mathbb{R}^3 \to \mathbb{R},\ (s', c, s) \mapsto s$$

Figure 3: The physical decomposition of the unmanned aerial vehicle (UAV), where d denotes the desired state, s' the predicted state, c the control action, s the current state, and e the environmental inputs. The functions coincide with U (Figure 2).

measurement y_p and uses it to create an estimate of the full state at that time s_{p+1}, and then uses that state information to update its control signal.

At this point, we have mapped a given behavioral formalism to our wiring diagram syntax and semantics for illustrative purposes. We posit that such diagrams can fully capture behaviors that adhere to composition. A different formalism could model the behavior of an information technology (IT) system by mapping a graph formalism used to assess and measure its performance. However, the usefulness of the formalism lies precisely in relating the physical behavior of the cyber-physical systems to its candidate architecture and therefore composition in this sense better captures the possible misbehavior a system designer might want to mitigate against.

We now wish to use the compositional system-as-algebra perspective to, in a sense, reverse engineer: given a specific behavioral description of some UAV we would like to provide specific descriptions for the individual component systems that would result in such a composite behavior. We will use the system model for predicting aircraft pitch [19]. In that specific example, the longitudinal equations of motion for a fixed-winged aircraft are given as a state-space model of the following form,

$$\begin{pmatrix} \dot{a} \\ \dot{q} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} -0.313 & 56.7 & 0 \\ -0.0139 & -0.426 & 0 \\ 0 & 56.7 & 0 \end{pmatrix} \begin{pmatrix} a \\ q \\ \theta \end{pmatrix} + \begin{pmatrix} 0.232 \\ 0.0203 \\ 0 \end{pmatrix} \begin{pmatrix} \delta \end{pmatrix} \tag{5}$$

$$y = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} a \\ q \\ \theta \end{pmatrix}$$

where a is the angle of attack, q is the pitch rate, θ is the pitch angle and δ is the deflection angle.

Working in the linear time-invariant system algebra (3), suppose $(S_{\mathtt{L}}, \mathscr{A}_{\mathtt{L}}, \mathscr{B}_{\mathtt{L}}, \mathscr{C}_{\mathtt{L}})$, $(S_{\mathtt{C}}, \mathscr{A}_{\mathtt{C}}, \mathscr{B}_{\mathtt{C}}, \mathscr{C}_{\mathtt{C}})$

and $(S_\mathtt{D},\mathscr{A}_\mathtt{D},\mathscr{B}_\mathtt{D},\mathscr{C}_\mathtt{D})$ are three linear systems inhabiting the respective boxes of Figure 3, with

$$\begin{aligned}
u_\mathtt{L}(s_\mathtt{L},e,s) &= \mathscr{A}_\mathtt{L}\cdot s_L+\mathscr{B}_L\cdot(e,s) & \qquad r_\mathtt{L}(s_\mathtt{L}) &= \mathscr{C}_\mathtt{L}\cdot s_\mathtt{L}\\
u_\mathtt{C}(s_\mathtt{C},d,s') &= \mathscr{A}_\mathtt{C}\cdot s_\mathtt{C}+\mathscr{B}_\mathtt{C}\cdot(d,s') & \qquad r_\mathtt{C}(s_\mathtt{C}) &= \mathscr{C}_\mathtt{C}\cdot s_\mathtt{C}\\
u_\mathtt{D}(s_\mathtt{D},c) &= \mathscr{A}_\mathtt{D}\cdot s_\mathtt{D}+\mathscr{B}_\mathtt{D}\cdot(c) & \qquad r_\mathtt{D}(s_\mathtt{D}) &= \mathscr{C}_\mathtt{D}\cdot s_\mathtt{D}
\end{aligned}$$

Using the algebra machinery (4), we can compute the composite linear dynamical system that inhabits the box `UAV`: its state space is $S_\mathtt{L}\times S_\mathtt{C}\times S_\mathtt{D}$, and its update and readout linear functions are

$$\begin{aligned}
u_\mathtt{UAV}\colon\ & S_\mathtt{L}\times S_\mathtt{C}\times S_\mathtt{D}\times\mathbb{R}^2\to S_\mathtt{L}\times S_\mathtt{C}\times S_\mathtt{D},\\
& (s_\mathtt{L},s_\mathtt{C},s_\mathtt{D},d,e)\mapsto\left(\mathscr{A}_\mathtt{L}s_\mathtt{L}+\mathscr{B}_\mathtt{L}\begin{pmatrix}e\\ \mathscr{C}_\mathtt{D}s_\mathtt{D}\end{pmatrix},\mathscr{A}_\mathtt{C}s_\mathtt{C}+\mathscr{B}_\mathtt{C}\begin{pmatrix}d\\ \mathscr{C}_\mathtt{L}s_\mathtt{L}\end{pmatrix},\mathscr{A}_\mathtt{D}s_\mathtt{D}+\mathscr{B}_D\mathscr{C}_\mathtt{C}s_\mathtt{C}\right)\\
r_\mathtt{UAV}\colon\ & S_\mathtt{L}\times S_\mathtt{C}\times S_\mathtt{D}\to\mathbb{R}\\
& (s_\mathtt{L},s_\mathtt{C},s_\mathtt{D})\mapsto\mathscr{C}_\mathtt{D}s_\mathtt{D}
\end{aligned}$$

We assume, for simplicity[4], that the state spaces of the sensor and controller are $\mathbb{R}$. Knowing that only the dynamics `D` relate to the triplet (a,q,θ), we set $S_\mathtt{D}=\mathbb{R}^3$ producing a resulting state space of the composite system $S_\mathtt{UAV}=\mathbb{R}^5$. Moreover, from the shape of the boxes we deduce that matrices $\mathscr{A}_\mathtt{L}$, $\mathscr{A}_\mathtt{C}$, $\mathscr{C}_\mathtt{L}$ and $\mathscr{C}_\mathtt{C}$ are one-by-one, $\mathscr{B}_\mathtt{L}$ and $\mathscr{B}_\mathtt{C}$ are one-by-two, whereas $\mathscr{A}_\mathtt{D}$ is three-by-three, $\mathscr{B}_\mathtt{D}$ is three-by-one and $\mathscr{C}_\mathtt{D}$ is one-by-three.

Decoding the above equations, we first of all have that the only output of the composite system is the output of `D`, namely

$$\mathscr{C}_\mathtt{UAV}=\begin{pmatrix}0 & 0 & {}_1(\mathscr{C}_\mathtt{D})_3\end{pmatrix}$$

Hence for obtaining equation (5), in the specific example we deduce that $\mathscr{C}_\mathtt{D}=\begin{pmatrix}0 & 0 & 1\end{pmatrix}$ meaning only θ is outputted to the outside world as desired.

Regarding the update part for an element of the state space $\mathbb{R}^5$ of the form $(s_\mathtt{L},s_\mathtt{C},\overbrace{a,q,\theta}^{s_\mathtt{D}})$, isolating the first two variables we obtain

$$\overset{\bullet}{s}_\mathtt{L}=\mathscr{A}_\mathtt{L}s_\mathtt{L}+{}_1(\mathscr{B}_\mathtt{L})_2\begin{pmatrix}e\\ \underbrace{\mathscr{C}_\mathtt{D}s_\mathtt{D}}_{\theta}\end{pmatrix}\qquad\overset{\bullet}{s}_\mathtt{C}=\mathscr{A}_\mathtt{C}s_\mathtt{C}+{}_1(\mathscr{B}_\mathtt{C})_2\begin{pmatrix}d\\ \mathscr{C}_\mathtt{L}s_\mathtt{L}\end{pmatrix}$$

which could be viewed as the 'extra info' of the composite system relating to the behaviors of the sensor and controller, not appearing in equation (5) but part of the total system's behavior. Now isolating the last three variables we obtain a description

$$\begin{pmatrix}\overset{\bullet}{\alpha}\\ \overset{\bullet}{q}\\ \overset{\bullet}{\theta}\end{pmatrix}={}_3(\mathscr{A}_\mathtt{D})_3\begin{pmatrix}\alpha\\ q\\ \theta\end{pmatrix}+{}_3(\mathscr{B}_\mathtt{D})_1\mathscr{C}_\mathtt{C}s_\mathtt{C}$$

Comparing with the desired equation (5), the deflection angle δ is the output of the controller $\mathscr{C}_\mathtt{C}s_\mathtt{C}$ which matches the physical reality, and the $\mathscr{A}_\mathtt{D}$, $\mathscr{B}_\mathtt{D}$ are completely determined by the composite description,

[4] In the likely case that sensor and controller are linear functions, to express them as linear time-invariant systems the state space would match the input space and therefore be $\mathbb{R}^2$. Moreover $\mathscr{A}$ would be the zero matrix and $\mathscr{B}$ would be the unit matrix.

namely

$$\mathscr{A}_{\mathtt{D}} = \begin{pmatrix} -0.313 & 56.7 & 0 \\ -0.0139 & -0.426 & 0 \\ 0 & 56.7 & 0 \end{pmatrix} \qquad \mathscr{B}_{\mathtt{D}} = \begin{pmatrix} 0.232 \\ 0.0203 \\ 0 \end{pmatrix}$$

The remaining data $\mathscr{A}_{\mathtt{L,C}}, \mathscr{B}_{\mathtt{L,C}}, \mathscr{C}_{\mathtt{L,C}}$ depend on engineering and physical parameters. Imposing extra, realistic conditions on what these could be is part of ongoing work.

In conclusion, this reverse engineering process is categorically described as follows. Given the algebra LTIS, a wiring diagram $f\colon \mathtt{L}\otimes\mathtt{C}\otimes\mathtt{D}\to\mathtt{U}$ in $\mathbf{W_{Lin}}$ (Figure 3) as well as an object of the target category $\mathrm{LTIS}(\mathtt{U})$, namely a specific linear system as in equation (5) inhabiting the outside box $\mathtt{UAV}$, the goal is to find an object in the pre-image of the given system under the composite functor

$$\mathrm{LTIS}(\mathtt{L})\times\mathrm{LTIS}(\mathtt{C})\times\mathrm{LTIS}(\mathtt{D}) \xrightarrow{\phi_{\mathtt{L,C,D}}} \mathrm{LTIS}(\mathtt{L}\otimes\mathtt{C}\otimes\mathtt{D}) \xrightarrow{\mathrm{LTIS}(f)} \mathrm{LTIS}(\mathtt{U}). \tag{6}$$

Clearly it is not expected that such a problem has a unique solution, but for example, in this specific case the system

$$(S_{\mathtt{D}}, \mathscr{A}_{\mathtt{D}}, \mathscr{B}_{\mathtt{D}}, \mathscr{C}_{\mathtt{D}}) \tag{7}$$

was completely determined by the composite system. Further work would aim to shed light on possible shapes of wiring diagrams that have better-behaved solutions under algebras of interest.

4 Decomposing Hierarchically to an Implementation

We have shown how to model a behavioral response of the system as it pertains to its physics. But how might we connect this behavioral understanding to a concrete implementation, in particular how can we zoom-in on each of the boxes to create an *architectural implementation* that will yield the wanted behaviors? Of course, there is no one solution for how to implement a cyber-physical system, so how might we categorically capture different architectural solution with an explicit relationship to the wanted set of behaviors?

Starting with a cyber-physical system from an engineer or designer point of view we now might want to model a candidate implementation. That means decomposing it into certain sub-components and using a specific wiring between them, following some choices based on the physical reality, experience, purpose and access to particular components at the time. Having formalized arbitrary zoomed-in pictures of a system or even more abstractly an agnostic process interface where various descriptions could live on using the category of labeled boxes and wiring diagrams, gives us the chance to view the above process using the basic notion of a *slice category* under an appropriate perspective.

Broadly speaking, for any category $\mathbf{C}$ and a fixed object $c\in\mathbf{C}$, we here think of $\mathbf{C}/c$ as containing all possible 'design choices' for c available to a system engineer; this formally models the possibility of implementing a system c in multitudes of ways. For example, in our setting a designer might choose to implement the flight control system – a subsystem of the UAV – either using a microcontroller or a field programmable gate array (FPGA). Similarly, the same applies to the rest of the system: there are a number of options for implementing the sensory system or dynamics.

In fact, we have already used a chosen implementation in the previous section. In more detail, suppose we have a process inhabiting some box $\substack{\mathbb{R}\\ \mathbb{R}}\boxed{\mathtt{U}}\,\mathbb{R}$, which is an object of $\mathbf{W}$. How can we decompose it into sub-processes, and how should they be interconnected to form the given system? All the possible decompositions can be thought of as the objects of the slice category $\mathbf{W}/\substack{\mathbb{R}\\ \mathbb{R}}\boxed{\mathtt{U}}\,\mathbb{R}$, therefore the architectural implementation is manifested by choosing one such object: for example, $f\colon \mathtt{L}\otimes\mathtt{C}\otimes\mathtt{D}\to\mathtt{U}$ is one such choice (Figure 1), namely a wiring diagram with target $\mathtt{U}$. Considering all possible subcomponents and wirings of a box could be thought of as the object-function of the functor $\mathbf{W}/(\text{-})\colon \mathbf{W} \to \mathbf{Cat}$.

To control a UAV we could implement a number of design solutions, each with different components and setup. For illustrative purposes we show one such candidate implementation, one that models a real UAV system. We do discuss possible deviations from this one design and how they are modeled in the slice category. For the design of a candidate implementation we use the following notational abbreviations:

$$\begin{aligned}
\mathtt{I} &\triangleq \text{ inertial measurement unit,}\\
\mathtt{P} &\triangleq \text{ processor,}\\
\mathtt{V} &\triangleq \text{ servos,}\\
\mathtt{X} &\triangleq \text{ aileron,}\\
\mathtt{Y} &\triangleq \text{ rudder,}\\
\mathtt{Z} &\triangleq \text{ throttle,}\\
\mathtt{W} &\triangleq \text{ elevator, and}\\
\mathtt{F} &\triangleq \text{ airframe.}
\end{aligned}$$

For example, a possible set of components for the implementation of the sensory system for a flight control system includes a gyroscope, an accelerometer, and a pressure sensor, which are usually realized through an inertial measurement unit (IMU). Because an IMU has accumulative error it is important to either add redundancy (that is, use two IMU devices) or add a separate device (for example, a global positioning unit (GPS) to ping and correct this error in IMU measurements). Current UAV systems include further functionality, for example, by using differential pressure sensors or adding a magnetometer to achieve further robustness during guided flight. Here we will focus on one possible design of the UAV, including implementation for the dynamics through control surfaces; that is, aileron, rudder, throttle, and elevator, and airframe. To manipulate the control surfaces we include servos, which are small motors attached to the control surfaces that move them based on the control laws that are implemented in the controller.

One of the important advantages of expressing system decompositions as a morphism in the category $\mathbf{W}$ is that we can perform further zoomed-in decompositions as desired, in a *hierarchical* way. For example, we may choose to implement the sensor box $\mathtt{L}$ using two IMU units $\mathtt{I}_1, \mathtt{I}_2$ and a processor $\mathtt{P}_1$ in a certain interconnection. Expressing this as a morphism with target $\mathtt{L}$, namely $g\colon \mathtt{I}_1 \otimes \mathtt{I}_2 \otimes \mathtt{P} \to \mathtt{L}$ means that we can compose this with the earlier f to obtain a two-level zoomed-in decomposition

$$(\mathtt{I}_1 \otimes \mathtt{I}_2 \otimes \mathtt{P}_1) \otimes \mathtt{C} \otimes \mathtt{D} \xrightarrow{g\otimes 1\otimes 1} \mathtt{L} \otimes \mathtt{C} \otimes \mathtt{D} \xrightarrow{f} \mathtt{UAV}.$$

that only 'opens-up' the box $\mathtt{L}$. We could moreover implement the control as well as the dynamics box, and decompose them in a choice of subcomponents and wires between them. An example where the control box is decomposed into $\mathtt{P}_2$ followed by $\mathtt{V}$ in a serial composition, and the dynamics box is decomposed into four parallel boxes, $\mathtt{X}$, $\mathtt{Y}$, $\mathtt{Z}$ and $\mathtt{W}$ followed by $\mathtt{F}$ amounts to choosing $h\colon \mathtt{P}_2 \otimes \mathtt{V} \to \mathtt{C}$ in $\mathbf{W}/\!-\!\boxed{\mathtt{C}}\!-$ and $k\colon \mathtt{X} \otimes \mathtt{Y} \otimes \mathtt{Z} \otimes \mathtt{W} \otimes \mathtt{F} \to \mathtt{D}$ in $\mathbf{W}/\!-\!\boxed{\mathtt{D}}\!-$. Combining all these morphisms we have the composition (Figure 4):

$$(\mathtt{I}_1 \otimes \mathtt{I}_2 \otimes \mathtt{P}_1) \otimes (\mathtt{P}_2 \otimes \mathtt{V}) \otimes (\mathtt{X} \otimes \mathtt{Y} \otimes \mathtt{Z} \otimes \mathtt{W} \otimes \mathtt{F}) \xrightarrow{g\otimes h\otimes k} \mathtt{L} \otimes \mathtt{C} \otimes \mathtt{D} \xrightarrow{f} \mathtt{UAV}$$

that can be considered as a single morphism from the tensor of all second-level sub-components to `UAV`, namely erasing the intermediate colored boxes.

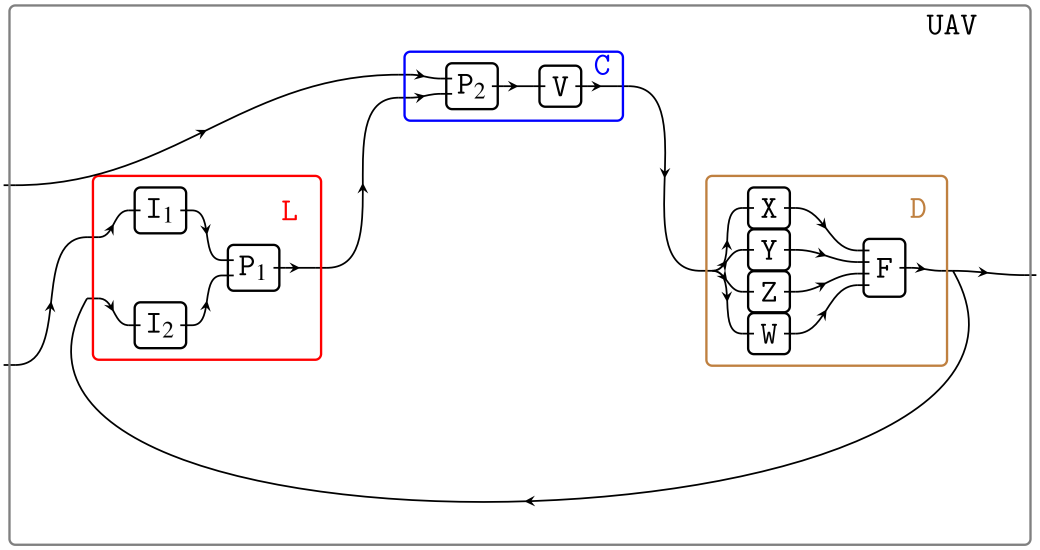


Figure 4: The component systems are further decomposed to candidate implementations.

A categorical approach to systems engineering allows us to hierarchically decompose from the system function to a candidate implementation while making explicit the traceability between those two related by different views of system design. Broadly speaking, this section's narrative relates to the architectural implementation of our systems, formally expressed by choosing the constituent boxes (domain) and their wiring interconnection (morphism) of a given system inside the category $\mathbf{W}$. The implementation comes together with the physical behavior of the UAV (Section 3) via the algebra structure of any lax monoidal functor $B\colon \mathbf{W} \to \mathbf{Cat}$, in this instance LTIS (3), as follows. For *any* `UAV`-implementation $f\colon \boxed{A} \to \boxed{\mathtt{UAV}}$, the functorial assignment

$$\begin{tikzcd}
B\colon \mathbf{W} \arrow[r] & \mathbf{Cat} \\
\boxed{A} \arrow[r, mapsto] \arrow[d, "f"'] & B(\boxed{A}) \arrow[d, "B(f)"] \\
\boxed{\mathtt{UAV}} \arrow[r, mapsto] & B(\boxed{\mathtt{UAV}})
\end{tikzcd}$$

allows us to compose a given B-system on $\boxed{A}$ to produce a B-system on $\boxed{\mathtt{UAV}}$, or more interestingly go backwards: given a B-system on `UAV`, we first choose an implementation $f\colon A \to \mathtt{UAV}$ where the domain could be the tensor of a number of sub-boxes, and then we choose an object of the system's pre-image similarly to the composite functor (6). This is a two-step process which appears in practice and depends on the choices of the system engineer, but can now be understood categorically through functorial semantics.

In particular, earlier the linear system (7) inhabiting the box $\boxed{\mathtt{D}}$ was completely determined by the equations of the composite system, whereas now we could further elaborate on the possible linear systems inhabiting the boxes X, Y, Z, W and F. Moreover, if such possibilities do not agree with physical intuition or the expected service of the system, such an implementation may be ruled out and alternatives identified appropriately based on the allowable set of behaviors.

5 Related Work

Systems theory has as long tradition and spans a large number of subfields, some more formal than others. A formal framework for systems theory was first built to describe biological systems [3, 28]. Since then, there have been several developments to systems theory that have assisted in areas such as control [21], safety assessment [17, 18], and security analysis [7, 29]. While we have been heavily influenced by systems theory, in this paper we develop a categorical approach to synthesizing and analyzing cyber-physical system designs on top of systems theory. This is because category theory is flexible with respect to relating different abstraction levels, which are necessary for the synthesis and analysis of cyber-physical systems.

The idea of using category theory for unifying specification languages is not new [9, 14]. However, we note that previous work in this area does not provide a concrete example of the specified system and it has also been applied predominantly for software engineering. Cyber-physical systems require modeling both the dynamics of the system and their implementation in software and hardware, which is an approach we illustrate in this paper. Further, such approaches to systems theory are old enough to require presenting again with the developments both in the theory and engineering of systems. While the categorical language can potentially be used for any system, we focus on its effectiveness to unify and scale models of cyber-physical systems, which pose new challenges because they operate in the environment.

In engineering too there has been research that takes advantage of category theory. A categorical formulation of hybrid systems that unifies all its views was presented by Ames and Sastry [2]. Our work views dynamical systems at a different abstraction than Ames and Sastry. We formalize the different views of a cyber-physical system as it would be modeled by systems engineers instead of focusing to the different topologies studied in control, which is often the case when designing systems in practice. Furthermore, Hasuo presented the idea of a research program for unifying the different views of cyber-physical systems using category theory [15]. Our work presents one (partial) solution to the problem outlined by Hasuo.

More recently, there have been other directions in the area of a categorical language for system design and assessment [5, 6], control [4], and even requirements management [11]. We view these research directions as complementary to the one presented in this paper. We present a framework for cyber-physical systems, which requires relating system assessment, modeling control, and refining requirements.

6 Conclusion

Applied category theory is a new field with promising solutions to the formal unification and dissemination of engineering design artifacts. This unification is of particular importance for designing and assuring correctness of cyber-physical systems, where misbehavior can lead to hazards. Cyber-physical system models require a number of views to provide evidence of correctness in their eventual deployment. We have shown that it is possible to create functorial relationships between the two main views of a cyber-physical system, the simulation of the desired behavior of the system and its decomposition to a specific candidate implementation. By building this functorial relationship it is possible to find consistency errors between those different model views and, therefore, better address misbehavior early in the system's lifecycle.

This is a first step to an eventual unification not only of methods in cyber-physical systems theory but also of tools. This would be an important accomplishment given that today systems are designed predominantly through tools. For example, tools are used to tune the control laws, simulate the overall system state, provide guarantees that the system does not reach death states, and compare and contrast different candidate implementations. We posit that working compositionally by formalizing cyber-physical systems modeling in category theory will in the future lead to scalable models of those complex systems. In addition, applied category theory has already shown positive results in the area of software execution, for example by using dependent types. Producing a cyber-physical systems modeling language blueprint in category theory could lead to having a fully integrated model that could be used throughout the lifecycle. In the beginning this model could be used to simulate the dynamical response and towards the end it could be used to synthesize code from the model, which would bridge the gap between system specification and implementation.

Beyond our general goals for this research program, our main immediate improvements to this work are to integrate discrete and continuous time, extend notions from contract-based design [22] in category-theoretic terms, and finally construct a fully traceable compositional model between requirements, behaviors, and implementations of cyber-physical systems.

7 Acknowledgments

G. Bakirtzis and C.H. Fleming are partially supported through the Systems Engineering Research Center (SERC) under USDOD Contract HQ0034-13-D-0004, and through the Northrop Grumman Mission Systems' University Research Program. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the USDOD nor Northrop Grumman.

C. Vasilakopoulou would like to thank the General Secretariat for Research and Technology (GSRT) and the Hellenic Foundation for Research and Innovation (HFRI).

References

[1] R. Alur (2015): *Principles of Cyber-Physical Systems*. The MIT Press.

[2] A. D. Ames & S. Sastry (2005): *A homology theory for hybrid systems: Hybrid homology*. In: *International Workshop on Hybrid Systems: Computation and Control*, Springer, doi:10.1007/978-3-540-31954-2_6.

[3] W. R. Ashby (1991): *Principles of the self-organizing system*. In: *Facets of systems science*, Springer, doi:10.1007/978-1-4899-0718-9_38.

[4] J. C. Baez & J. Erbele (2015): *Categories in control*. *Theory and Applications of Categories*. Available at http://www.tac.mta.ca/tac/volumes/30/24/30-24.pdf.

[5] S. Breiner, O. Marie-Rose, B. S. Pollard & E. Subrahmanian (2019): *Operadic diagnosis in hierarchical systems*. In: *Proceedings of the 2019 Applied Category Theory Conference*. Available at http://www.cs.ox.ac.uk/ACT2019/preproceedings/Spencer%20Breiner,%20Olivier%20Marie-Rose,%20Blake%20Pollard%20and%20Eswaran%20Subrahmanian%20Operadic%20diagnosis%20in%20hierarchical%20systems.pdf.

[6] S. Breiner, R. D. Sriram & E. Subrahmanian (2019): *Compositional Models for Complex Systems*. In: *Artificial Intelligence for the Internet of Everything*, Elsevier, doi:10.1016/B978-0-12-817636-8.00013-2.

[7] B. T. Carter, G. Bakirtzis, C. R. Elks & C. H. Fleming (2018): *A systems approach for eliciting mission-centric security requirements*. In: *Proceedings of the 2018 Annual IEEE International Systems Conference (SysCon)*, IEEE, doi:10.1109/SYSCON.2018.8369539.

[8] F. Durán, R. Heinrich, D. Pérez-Palacín, C. L. Talcott & S. Zschaler (2020): *Composing Model-Based Analysis Tools (Dagstuhl Seminar 19481)*. In: *Dagstuhl Reports*, Schloss Dagstuhl-Leibniz-Zentrum für Informatik. Available at `https://drops.dagstuhl.de/opus/volltexte/2020/11985/`.

[9] J. L. Fiadeiro & T. Maibaum (1995): *Interconnecting formalisms: supporting modularity, reuse and incrementality*. In: *Proceedings of the 3rd ACM SIGSOFT Symposium on Foundations of Software Engineering*, doi:10.1145/222124.222141.

[10] B. Fong & D. I. Spivak (2019): *An Invitation to Applied Category Theory: Seven Sketches in Compositionality*. Cambridge University Press.

[11] S. Gebreyohannes, W. Edmonson & A. Esterline (2018): *Formalization of the responsive and formal design process using category theory*. In: *Proceedings of the 2018 Annual IEEE International Systems Conference (SysCon)*, IEEE, doi:10.1109/SYSCON.2018.8369508.

[12] M. Gell-Mann (1994): *Complex adaptive systems*. *Complexity: Metaphors, Models, and Reality*. Available at `https://authors.library.caltech.edu/60491/`.

[13] M. Gell-Mann & S. Lloyd (2010): *Effective complexity*. In: *Murray Gell-Mann: Selected Papers*, World Scientific, doi:10.1142/9789812836854_0027.

[14] J. A. Goguen (1991): *A categorical manifesto*. *Mathematical structures in computer science*, doi:10.1017/S0960129500000050.

[15] I. Hasuo (2017): *Metamathematics for systems design*. *New Generation Computing*, doi:10.1007/s00354-017-0023-1.

[16] N. Kibret, W. W. Edmonson & S. Gebreyohannes (2019): *Category Theoretic Based Formalization of the Verifiable Design Process*. In: *Proceedings of the 2019 IEEE International Systems Conference (SysCon 2019)*, IEEE.

[17] N. G. Leveson (2004): *A new accident model for engineering safer systems*. *Safety science*, doi:10.1016/S0925-7535(03)00047-X.

[18] N. G. Leveson (2017): *Rasmussen's legacy: A paradigm change in engineering for safety*. *Applied ergonomics*, doi:10.1016/j.apergo.2016.01.015.

[19] B. Messner, D. Tilbury, R. Hill & J. D. Taylor (2020): *Control Tutorials for Matlab and Simulink: Aircraft Pitch*. `https://web.archive.org/web/20200509164711/http://ctms.engin.umich.edu/CTMS/index.php?example=AircraftPitch&section=SystemModeling`.

[20] R. Rajkumar, I. Lee, L. Sha & J. Stankovic (2010): *Cyber-physical systems: the next computing revolution*. In: *Proceedings of the Design Automation Conference*, IEEE, doi:10.1145/1837274.1837461.

[21] J. Rasmussen (1985): *The role of hierarchical knowledge representation in decisionmaking and system management*. *IEEE Transactions on systems, man, and cybernetics*, doi:10.1109/TSMC.1985.6313353.

[22] A. L. Sangiovanni-Vincentelli, W. Damm & R. Passerone (2012): *Taming Dr. Frankenstein: Contract-Based Design for Cyber-Physical Systems*. *European Journal of Control*, doi:10.3166/ejc.18.217-238.

[23] M. Saravi, L. Newnes, A. R. Mileham & Y. M. Goh (2008): *Estimating cost at the conceptual design stage to optimize design in terms of performance and cost*. In: *Proceedings of the 15th ISPE International Conference on Concurrent Engineering*, Springer, doi:10.1007/978-1-84800-972-1_11.

[24] P. Schultz, D. I. Spivak & C. Vasilakopoulou (2020): *Dynamical systems and sheaves*. *Applied Categorical Structures*, doi:10.1007/s10485-019-09565.

[25] D. I. Spivak (2016): *The steady states of coupled dynamical systems compose according to matrix arithmetic*. ArXiv:1512.00802 [math.CT].

[26] D. I. Spivak (2020): *Poly: An abundant categorical setting for mode-dependent dynamics*. ArXiv:2005.01894 [math.CT].

[27] D. Vagner, D. I. Spivak & E. Lerman (2015): *Algebras of open dynamical systems on the operad of wiring diagrams*. Theory and Applications of Categories. Available at `http://www.tac.mta.ca/tac/volumes/30/51/30-51.pdf`.

[28] L. Von Bertalanffy (1950): *An outline of general system theory.* British Journal for the Philosophy of science, doi:https://doi.org/10.1093/bjps/I.2.134.

[29] W. Young & N. G. Leveson (2014): *An integrated approach to safety and security based on systems theory*. Communications of the ACM, doi:10.1145/2556938.