Department of Computer Science and Engineering

Sheaf Semantics for Physically Motivated Network Description with Applications

Kari Visala

Aalto University

SCIENCE +
TECHNOLOGY

TECHNICAL REPORT

Aalto University publication series
SCIENCE + TECHNOLOGY 10/2014

Sheaf Semantics for Physically Motivated Network Description with Applications

Kari Visala

Aalto University
School of Science
Department of Computer Science and Engineering

Aalto University publication series
SCIENCE + TECHNOLOGY 10/2014

ISBN 978-952-60-5794-1 (pdf)
ISSN-L 1799-4896
ISSN 1799-4896 (printed)
ISSN 1799-490X (pdf)
http://urn.fi/URN:ISBN:978-952-60-5794-1

Unigrafia Oy
Helsinki 2014

Finland

Aalto University

Abstract

Aalto University, P.O. Box 11000, FI-00076 Aalto www.aalto.fi

Author
Author(s): Kari Visala

Name of the publication
Sheaf Semantics for Physically Motivated Network Description with Applications

Publisher School of Science

Unit Department of Computer Science and Engineering

Series Aalto University publication series SCIENCE + TECHNOLOGY 10/2014

Field of research Network and protocol modeling

Abstract

This paper introduces the notion of event space, a physically motivated mathematical model of distributed non-deterministic concurrent interaction based on Goguen's work on sheaf semantics. It provides unifying compositional semantics suitable for network and protocol description, which we demonstrate by designing a formal language, network resource calculus (NRC), for which we use the event space as a model. We sketch how NRC can be utilized in two application scenarios: specification of information-centric protocols and proof-carrying network description in protocols, that can be used together.

Keywords information-centric networking, network description language, sheaf, semantics, protocol specification

ISBN (printed) **ISBN (pdf)** 978-952-60-5794-1

ISSN-L 1799-4896 **ISSN (printed)** 1799-4896 **ISSN (pdf)** 1799-490X

Location of publisher Helsinki **Location of printing** Helsinki **Year** 2014

Pages 11 **urn** http://urn.fi/URN:ISBN:978-952-60-5794-1

Sheaf Semantics for Physically Motivated Network Description with Applications

Kari Visala
Aalto University School of Science
Email: kari.visala@aalto.fi

***Abstract*—This paper introduces the notion of *event space*, a physically motivated mathematical model of distributed nondeterministic concurrent interaction based on Goguen's work on sheaf semantics. It provides unifying compositional semantics suitable for network and protocol description, which we demonstrate by designing a formal language, *network resource calculus* (NRC), for which we use the event space as a model. We sketch how NRC can be utilized in two application scenarios: specification of information-centric protocols and proof-carrying network description in protocols, that can be used together.**

I. INTRODUCTION

Denotational semantics for network or protocol specification languages is the formalization of their meaning by assigning mathematical objects called denotations for the expressions of the languages. Our goal here is to introduce a mathematical structure for generic concurrent, distributed, nondeterministic interacting processes that would provide a particularly suitable model for languages describing communication protocols and networks. We chose a *physically motivated* approach that includes more details than is usual for models of concurrency [1]. For example, Internet is a global network and its nodes are geographically dispersed, which makes delay often an essential issue. Different physical layer transmission media have different characteristics that have direct consequences on what is possible on the higher layers, protocols may have real-time requirements, nodes can be mobile etc. If the model can closely reflect the structure of the actual system, it can provide modular building blocks that can be easily reused in applications. We manage the added complexity stemming from the modeling of physical aspects by first fixing a single structure that can express all these aspects explicitly and then using it to give model-theoretic semantics for a theory in first-order logic, that can be used as a specification language allowing flexible abstraction of unwanted details.

Due to the space constraints, the reader is assumed to be familiar with the basic concepts of category theory [2] and the definitions related to topological spaces. Here the terms "diagram", "limit", and "cone" are used in the sense of category theory. In section II we concisely introduce sheaves and summarize the work of Goguen in sheaf semantics [3]. The main contribution of this paper is the introduction of the physically motivated mathematical model of networks changing in time based on sheaves in section III and the design of a formal network specification language with sheaf semantics in section IV. We briefly explain how this basis can be used for two synergetic applications in section V.

II. MODELING OF CONCURRENT OBJECTS BY SHEAVES

The notion of a *sheaf* [4] describes mathematical structures with certain desired properties on a topological space. Firstly, *presheaves* are structures, which are required to have a way to restrict the structure $\mathcal{F}(U)$ defined on all of the open subsets U of a topological space X to the structure $\mathcal{F}(U)|_V$ on open subsets $V \subseteq U$. Secondly, if $\mathcal{F}(U)$ can be consistently and uniquely pieced back together (collated) from its restrictions to the subsets V_i of any open cover of U, the structure is a sheaf. Below, we will give the technical definition of this restriction-collation description that captures the exact requirements that allow global structures to be defined locally on X.

Definition 1: Let $\mathcal{T}(X)$ be a category, which has as its objects all the open subsets U of a topological space X and has an *inclusion* arrow $i : V \to U$ iff $V \subseteq U$. Here we assume $\mathcal{C}$ to be a complete category. A *presheaf* is a functor $\mathcal{F} : \mathcal{T}(X)^{op} \to \mathcal{C}$. Category $\mathcal{C}$ above is called the *structure category* of the presheaf. The arrows $\mathcal{F}(i)$ in $\mathcal{C}$ for $i : U \to V$ in $\mathcal{T}(X)$ are called the *restriction* morphisms induced by i.

Definition 2: A presheaf $\mathcal{F}$ is a *sheaf*, iff the following is an equalizer diagram in $\mathcal{F}$:

$$\begin{tikzcd}
\mathcal{F}(U) \arrow[r, dashed] & \prod_i \mathcal{F}(U_i) \arrow[r, shift left] \arrow[r, shift right] & \prod_{i,j} \mathcal{F}(U_i \cap U_j)
\end{tikzcd}$$

for all open coverings $U = \bigcup U_i$. The first arrow is the product of the restriction maps and the parallel arrows are the compositions of respective projections with restrictions. This is called the *sheaf condition*. If the index set i is limited to be finite, then we call it the *finite sheaf condition*.

Next we can define the category of $\mathcal{C}$-valued presheaves, which at the same time defines its full subcategory of sheaves. Let $\mathcal{F}$ and $\mathcal{F}'$ be two presheaves on topological space X with structure category $\mathcal{C}$. A *(pre)sheaf morphism* $\varphi : \mathcal{F} \to \mathcal{F}'$ is a family of maps $\varphi_U : \mathcal{F}(U) \to \mathcal{F}'(U)$, one for each open subset U of X such that for every $i : U \to V$ in $\mathcal{T}(X)$, the diagram below commutes in $\mathcal{C}$:

$$\begin{tikzcd}
\mathcal{F}(V) \arrow[r, "\varphi_V"] \arrow[d, "\mathcal{F}(i)"'] & \mathcal{F}'(V) \arrow[d, "\mathcal{F}'(i)"] \\
\mathcal{F}(U) \arrow[r, "\varphi_U"] & \mathcal{F}'(U)
\end{tikzcd}$$

It immediately follows from the definition that the presheaf morphisms are *natural transformations* of the presheaf functors forming the arrows of the functor category.

Goguen showed in [3] how concurrent, interacting objects give rise to sheaves. The approach is not limited to object-oriented systems, but can analogously model physical layer systems such as electrical circuits with continuous time that interact with different types of objects. Basically, an object is considered as a complete set of consistent observations on a topological space. Goguen defines a *preobject* $\mathcal{O}$ over a topological space X with *attribute object* $A = A_1 \times \cdots \times A_n$ to be a presheaf of the form $\mathcal{O}(U) = \{h : U \to A_1 \times \cdots \times A_n \mid K(h)\}$, where the morphisms $\mathcal{O}(i)$ are restriction maps, and K is a relation that expresses constraints that $\mathcal{O}$ satisfies. Here a single function h is interpreted as a consistent set of "observations" at different points in space and time and the set of such "possible worlds" forms a kind of phase space of the object. This description naturally supports non-deterministic computations, when there are more than one global h.

It follows from the functoriality of the presheaves that observations are closed under restriction, which is a generalization of the prefix closure condition of trace-based models. The system is "relational" in the sense, that it does not distinguish inputs from outputs, but the behavior of a system composed of connected objects is formed from the compatible behaviors of each object. This type of formulation is compositional and avoids the Brock-Ackerman anomaly [5].

More precisely, Goguen first showed that the objects give rise to a full subcategory $\mathcal{O}bj(X, \mathcal{C})$ of sheaves over a given topological space X and structure category $\mathcal{C}$. Then he introduced a specific type of object morphism called a *projection morhisms*. For two objects $\mathcal{O}$ and $\mathcal{O}'$ with attribute objects $A = \prod_{n \in N} A_n$ and $A' = \prod_{n \in N'} A_n$ respectively, such that $N' \subseteq N$, then $a : A \to A'$ sending $(a_n \mid n \in N)$ to $(a_n \mid n \in N')$ induces the components of a natural transformation, and thus, an object morphism. This can be interpreted as an simple form of *inheritance* of the common attributes from $\mathcal{O}'$ by $\mathcal{O}$. Non-projection morphisms can be thought to describe a kind of generalized inheritance that changes the representation of the object.

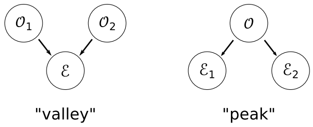

Fig. 1. In a "valley" diagram two objects inherit from a common third object for interaction. A "peak" diagram describes a relation between two "languages".

[3] describes how two objects $\mathcal{O}$ and $\mathcal{O}'$ can be composed to form interacting parts of a *system* by inheriting from a common third object $\mathcal{E}$, which is often a kind of shared "language" or "medium" consisting of all possible traces in the communication medium having states in some attribute objects. This forms a "valley" diagram depicted in Fig. 1. In the same figure, a "peak" diagram is depicted, where $\mathcal{O}$ can be thought to describe a relation between the two languages $\mathcal{E}_1$ and $\mathcal{E}_2$. These two examples can be generalized to the diagrams in the category $\mathcal{O}bj(X, \mathcal{C})$ that describe systems of connected objects. Finally, the behavior of a system is given by the limit of its diagram.

III. EVENT SPACES

The basic notion in our model is the *event space (ES)*. It describes the *behavior* of a process as a set of its possible *scenarios*, which are consistent sets of *events* that could occur together in some spacetime structure as the observable manifestation of the process. We can use this concept to model entities of varying size such as components of a router, an autonomous system (AS) of the Internet, or a software object.

Definition 3: Let X be a topological space and $\mathcal{T}(X)$ be a category defined as in Definition 1. A *C-valued pre-event space (pre-ES)* on X associates each open set U of X a set $\mathcal{D}(U)$ of *sections* on U of the form

$$\mathcal{D}(U) = \Big\{\{e \mid e : U \times C\}\Big\}$$

and to each open set $V \subseteq U$ a *restriction* map $r : \mathcal{D}(U) \to \mathcal{D}(V)$, which is defined $r(S) = \{(x, d) \mid (x, d) \in S \wedge x \in V\}$.

Events e are ordered pairs, where the first entry is a *point* of X that locates the event and the second entry $\pi_2(e)$ of type C is called the *data* of the event. From here on we will assume that C is $\mathbb{N}$. Sections of X are called scenarios and they are sets of events and sets of scenarios are called behaviors.

Theorem 3.1: A pre-ES $\mathcal{D}$ is a presheaf $\mathcal{D} : \mathcal{T}(X)^{op} \to \mathcal{S}et$, where $\mathcal{S}et$ [2] is the category of sets, which has sets as objects and total functions between sets as arrows. The composition of arrows in $\mathcal{S}et$ is the composition of functions.

Proof: If we have open subsets $V \subseteq U \subseteq T$ of X, then composition $r_2 \circ r_1 : \mathcal{D}(T) \to \mathcal{D}(V)$ of restriction maps $r_1 : \mathcal{D}(T) \to \mathcal{D}(U)$ and $r_2 : \mathcal{D}(U) \to \mathcal{D}(V)$ is $r(S) = \{(x, d) \mid (x, d) \in S \wedge x \in V\}$ which is the same as direct restriction from T to V, that is, restrictions are transitive when function composition is used. Also, all restrictions to the same open subset are trivially identity maps. These two properties are enough to guarantee the functoriality of $\mathcal{D}$. ■

A pre-ES is an ES, if it adheres to the sheaf condition. If the behavior on X of a (pre-)ES $\mathcal{D}$ contains an empty set, we call $\mathcal{D}$ a *possible ES*. In case the behavior on X is an empty set, we call $\mathcal{D}$ *impossible*. Sometimes we may want to model a relation between other types of processes and the events of an ES. For example, we may want to express the interface between the physical layer, that could be an electromagnetic field used for wireless communication, and link layer modeled as an ES. This can be achieved by using sheaves with sections of type $A \times E$, where A could model the structure of the physical layer and a projection morphism could be used to extract an ES with behavior E. We also note, that as the different scenarios can be interpreted to mean possible actual outcomes of a process, we can easily extend (pre-)ESes to a probabilistic model by assigning a *probability measure* to the scenarios. We envision this to be a natural way to model component failures and traffic conditions as (pre-)ESes. However, this is left as a future work.

We assume (pre-)ESes to be defined on the topology of some physical notion of *spacetime*. The first entry $\pi_1(e)$ locates an event e both in space and time. For example, the space X could be a Riemannian 4-manifold representing the Earth's surface, height, and time dimension or, for a practical representation of the points, GPS coordinates and discrete global time could be used as the coordinates of the events.

In order to model realistic digital systems, we define a more restricted form of ES called *discrete event space (DES).*

Definition 4: A (pre-)DES on topological space X is a (pre-)ES on X that has a finite number of events in all compact subsets of X in all of its scenarios.

This requirement for a DES implies the *discreteness* law in the *actor model of concurrency* [6]. For example, in Euclidean space $\mathbb{R}^n$ we can form a compact set by constructing a *convex hull* containing any finite number of events. It follows, that there can be no accumulation points of events in X and it is not possible to build "Zeno machines" that compute an infinite number of steps in a finite space and time [1]. It should be noted however, that just being a sheaf, a DES is conjectured by Goguen [3] to include only computable functions whereas presheaves correspond to arbitrary specifications. For example, the famous example of *fair merge* can be expressed as a presheaf but not as a sheaf [3]. The definition of DES also opens another direction of potential work, as it might be interesting to try to understand "physical" computational complexity in terms of used spacetime "resource".

A. Categories of Event Spaces

Natural transformation between pre-ESes form the morphisms of a functor category $pre\text{-}\mathcal{E}s(X)$ on a topological space X and structure category $\mathcal{S}et$. Restricting to morphisms between (D)ESes, the full subcategories $\mathcal{E}s(X)$ and $\mathcal{D}es(X)$ of $pre\text{-}\mathcal{E}s(X)$ are defined.

Following Goguen, we can interpret small diagrams in $pre\text{-}\mathcal{E}s(X)$ as *systems* of connected pre-ES "components" as explained in section II. The limit of a diagram for presheaves can be computed "pointwise". That is, if we have a diagram labeling event spaces $\mathcal{S}_n$ connected by morphisms $\varphi_e : \mathcal{S}_i \to \mathcal{S}_{i'}$ on a topological space X for indices n, e, i of a *small* diagram, then for the limit object $\mathcal{S}$ of the diagram, denoted here by $\lim S_n$, holds that $(\lim S_n)(U) = \lim(S_n(U))$ for each open subset U of X [7]. The existence of the limit is guaranteed in our case by the bicompleteness of the structure category $\mathcal{S}et$. In $\mathcal{S}et$, we can construct this limit for each U as the set of all *consistent nets of points* $\mathcal{L}(U)$ in S

$$\mathcal{L}(U) = \left\{ \{s_n\} \;\middle|\; \begin{array}{l} s_n \in S_n(U) \;\wedge (\varphi_{e,U} : S_{n'}(U) \to S_n(U) \;\wedge \\ \varphi_{e',U} : S_{n''}(U) \to S_n(U) \\ \;\Rightarrow \varphi_{e,U}(s_{n'}) = \varphi_{e',U}(s_{n''})) \end{array} \right\}$$

where $\varphi_{e,U}$ and $\varphi_{e',U}$ are the components of the natural transformations φ_e and $\varphi_{e'}$ at U. $\mathcal{L}(U)$ contains the largest set of consistent scenarios for all (pre-)ES in the diagram that makes the diagram commute. The diagram can be thought as a kind of equation of connections and the limit as the solution.

The interaction between pre-ESes is *relational* in the sense, that we do not distinguish inputs from outputs or assume an *arrow of time*: when we connect two pre-ESes, it is possible to interpret either one as being an input to the process represented by the other pre-ES. This is also justifiable by the CPT symmetry in physical laws. Even when all pre-ESes have been connected, the system may have more than one scenario, in which case we can interpret the system to be nondeterministic so that it may exhibit any of these scenarios when actually run. Another interpretations is to assume some pre-ESes to be unconnected "interfaces" of the system and model *open systems* that can be extended or given input, which is desirable when modeling systems such as the Internet, which is independently built by multiple parties. Pre-ESes form fully *compositional* pieces that can be given semantics independent of the system they are part of and the meaning of the system is a function of the meanings of its parts.

Pre-ESes might also be able to model *quantum computers* straightforwardly as the non-deterministic behavior of each pre-ES could be interpreted as a *wave function* and a system of pre-ESes corresponds to the interaction or *entanglement* of several entities restricting possible resulting states of each entity. The relational model of pre-ESes is compatible even with closed timelike curves (CTC) allowed by general relativity: pre-ESes can be connected "in a loop" following a CTC, but this is not different from other types of loops in connections as only consistent behaviors are accepted for the whole system. In this sense we could say that our model is compatible with the Novikov's self-consistency conjecture.

In many practical cases, we want to assume more structure for the topological space X. For example, a metric space allows us to model signal velocity in the network. Often we want to define a *causal relation* $\leq$ between events that forms a *preorder*, a relation that is both *reflexive* ($e \leq e$) and *transitive* ($a \leq b \wedge b \leq c \Rightarrow a \leq c$). Then we need a way to map this causal structure to the spacetime underlying the ES in a structure preserving way. For example, for time-orientable Lorentzian manifolds we could have for events e_1 and e_2 that $e_1 \leq e_2$ implies a *future-directed non-spacelike* curve from $\pi_1(e_1)$ to $\pi_1(e_2)$. For Minkowski space the $\leq$ becomes a partial order that corresponds to vector $\pi_1(e_2)$ - $\pi_1(e_1)$ being future directed null or future directed timelike. Partial order is antisymmetric ($a \leq b \wedge b \leq a \Rightarrow a = b$) and has no loops.

B. Using Morphisms to Describe Event Spaces

We can also use different kinds of projection morphisms in category $pre\text{-}\mathcal{E}s(X)$ to define new pre-ESes from known pre-ESes or partially describe hidden pre-ESes by giving some of their known projections as a diagram. We might want to define a router as a black box with certain known types of

---

[1] The DES condition seems to also rule out some models of *hypercomputation* based on "wormholes" appearing in general relativity as *closed timelike curves (CTC)* that form compact spaces: it is impossible to use the result of an infinite computation.

interfaces, that are pre-ESes. This way we can form partial descriptions of systems and compute their external behaviors at the interfaces while the implementations are unknown.

Definition 5: Let $\varphi : \mathcal{C} \rightarrow \mathcal{D}$ be a morphism in *pre-*$\mathcal{E}s(X)$. Also $\varphi_U(s) \subseteq s$ for each scenario s in $\mathcal{C}(U)$ for each open subset U of X. We call such morphisms *subbehavior morphisms (SB)*. If the components φ_U of φ are surjective functions, we say that $\mathcal{D}$ is a *subbehavior* of $\mathcal{C}$.

We can now define useful special cases of SBs. Firstly, an *identity projection* is an SB that maps any scenario to itself. An *event filter* (EF) $ef(f) : \mathcal{C} \rightarrow \mathcal{D}$ is an SB that maps each scenario s_c of $\mathcal{C}(U)$ to a scenario $\{e \mid e \in s_c \wedge f(\pi_2(e)) = 1\}$ of $\mathcal{D}(U)$ for some *subobject* $f : \mathbb{N} \rightarrow \Omega$ where the *subobject classifier* $\Omega = \{0, 1\}$ in $\mathcal{S}et$ and U is any open subset of X. EFs can be used to extract events matching the criteria represented by f from a more complicated pre-ES. For example, we could isolate certain protocol under study from the activity of a complex network represented as an ES, where events e_i correspond to sending and receiving of packets at locations $\pi_1(e_i)$ with contents $\pi_2(e_i)$.

Another useful SB for modeling physical components is a *subcomponent morphism (SC)*. An SC $sc(Y) : \mathcal{C} \rightarrow \mathcal{D}$ is a SB that maps each scenario s_c of $\mathcal{C}(U)$ to a scenario $\{e \mid e \in s_c \wedge \pi_1(e) \in Y\}$ of $\mathcal{D}(U)$ for some $Y \subseteq X$ and U is any open subset of X. If the mapping is surjective, way say that $\mathcal{D}$ is a *subcomponent* of $\mathcal{C}$. An SCs projects a typically localized part of the whole to a new pre-ES. We mostly use SCs to pick out external *interfaces* of an ES so that they can be connected to other ESes. For example, an ES $\mathcal{C}$ could have a subcomponent $input$ and another ES $\mathcal{D}$ a subcomponent $output$ (that is, there are SCs $i : \mathcal{C} \rightarrow input$ and $o : \mathcal{D} \rightarrow output$). Then we could connect $\mathcal{C}$ and $\mathcal{D}$ by forming a diagram with objects $\mathcal{C}$, $\mathcal{D}$, $input$, and $output$ and arrows i, o and identity projections from $output$ to $input$ connecting the interfaces to form a system. Alternatively, we could have added one more ES in the middle and connected both $input$ and $output$ to it. This additional ES could model some kind of "medium" of the connection and its scenarios would have determined the possible "states" of the medium such as maximum frequency of events that could be used to model network bandwidth, for example. If we want to model cable media with its ends at two different places, we need to model the cable as a separate ES with its own properties such as delay and both of the ends are connected to interfaces using SCs starting from the cable ES as explained above. Next we define two types of unidirectional i/o interface pre-DESes used in our examples that are typically connected using an identity projection from output to input interface:

Definition 6: A *Discrete event input interface (DEII)* and *Discrete event output interface (DEOI)* are pre-DESes on topological space X with the additional structure of causal preorder $\leq$ between the events in X. The behavior of a DEII for some subset $Y \subseteq X$ is defined as all subsets of $\{e_i \mid e_i \in Y \times \mathbb{N} \wedge \pi_1(e_i) = \pi_1(e_j) \Rightarrow \pi_2(e_i) = \pi_2(e_j)\}$.

In [8] we specified an *universal information interface (UII)* as a general API for publishing and receiving data values. Here we can give UII the semantics DEII × DEOI.

When we consider arbitrary morphisms $m : \mathcal{C} \rightarrow \mathcal{D}$ in category *pre-*$\mathcal{E}s(X)$, they can be considered as general *properties* or *interpretations* of $\mathcal{C}$ represented as another pre-ES on X and can be used for the analysis of $\mathcal{C}$. Such morphism can change the events of $\mathcal{C}$ and when modeling networks, we can construct a morphism that projects an *overlay network* of some kind from the network implementing it. We could also express the overlay as a diagram by abstracting the underlay into a single ES, but in this case the network topology of the overlay is not clearly visible, but the overlay nodes are connected to the underlay ES. *Layering*, *virtual private networks*, *tunnels*, and *scopes* [8] are examples of such overlays.

C. Time Evolution of Systems

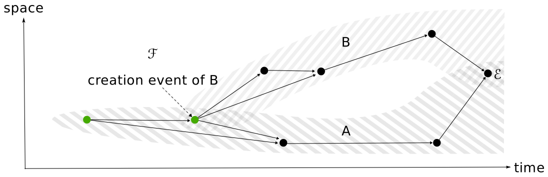

Fig. 2. Two consistent scenarios of ESes A and B are shown with their events depicted as dots and some of the causality relations as arrows. The creation event of B is synchronized with the creation event of B in A. At later time, A and B communicate at $\mathcal{E}$.

Compared to the Goguen's inheritance explained in section II, subcomponents have a clear physical interpretation as they can be localized in time and space. Because the location of an event in the topological space X is assumed to reflect its location in the modeled system, our approach automatically can describe as diagrams *dynamic networks* that change in time. For example, in Fig. 2 we have depicted two compatible scenarios of ESes A and B that do not coexists at all points in time. Our spacetime model is flexible as it extends naturally to relativistic spacetimes without absolute notion of simultaneity.

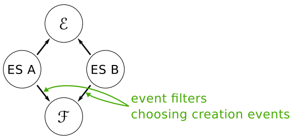

Fig. 3. All transitively potentially created ESes are included in (small) diagrams and creation events between them are synchronized by an additional mediating ES $\mathcal{F}$ connected as a valley. Here, the other valley $\mathcal{E}$ is used for ordinary interaction between the components.

We can also describe dependencies from the events to the creation of new components in the system. In Fig. 2 the creation event of ES B is shared by both ESes in compatible scenarios. Here the ES A creates a new ES B that is connected to it. This can be described as a diagram as shown in Fig.

3. As small diagrams can be infinite, we can simply add every possible ES created to it. Possible ESes have an empty scenario, which means that they will not manifest themselves in any way if they are not created. For example, we can assume that each non-empty scenario of the possible ES B has a creation event that is required to causally precede all other events of ES B. An additional ES $\mathcal{E}$ and event filters from the creator ES and the created ES to $\mathcal{E}$ can be used to synchronize the creation so that the created ES only exist in those scenarios where it was actually created. Notably this formulation does not assume an arrow of time but the synchronization can be based on any condition. The created ES can naturally interact with other ESes in the scenarios it is created.

IV. FORMAL NETWORK SPECIFICATION

So far we have interpreted pre-ESes with multiple scenarios as nondeterministic processes, open systems that can interact, or a mixture of these two. The behavior of a pre-ES can also be interpreted as a "collapsed"[2] specification of allowed *covariant* scenarios as explained below.

We restrict our attention to systems of pre-DESes, where the communication is modeled by attaching subcomponent DEOIs and DEIIs to a common medium pre-DES using identity projections while each DEOI is connected only to a single medium pre-DES. All medium pre-DESes have behaviors $2^{X\times\mathbb{N}}$ on topological space X. Each component pre-DES is required to have at least one scenario. If the component has *contravariant* input interfaces, it also needs to have one scenario for all possible combinations of input.

We introduce a formal language that uses first-order logic to define pre-DES system specifications. A system behavior is a set of consistent scenarios for all of the constituent pre-DESes as explained in section III. This can be attained by the limit of a diagram describing the connections between pre-DES components connected as a *network*. The idea is that we can define the set of allowed consistent global system scenarios, where the possible component scenarios are elements of the power set of all events $2^{X\times\mathbb{N}}$ on topological space X, as the models satisfying the first-order sentences interpreted as pre-DES system specifications. If all consistent scenarios s_i of a system behavior b satisfy a given sentence ϕ, $s_i \models \phi$ in our intended interpretation while having a scenario for all possible combinations of input, we say that b implements the specification ϕ, $b : \phi$. If a sentence β implies sentence α, we can interpret it as a stricter specification and say that β is a subtype of α, $\beta <: \alpha$. It should be noted that all global behaviors define pre-ESes as the restriction maps always exist. Even though such specifications may not generally be ESes, the subset of a pre-ES behavior can be an ES.

We can treat system specifications as concrete networks and connect them with new connections to form specifications of larger interconnected networks. If we remove some scenarios from some constituent parts of a network, no new consistent scenarios for the whole network are formed which leads to the following *system specification theorem*:

Theorem 4.1: If a part of a connected system with specification γ has specification α, which is replaced by a specification $\beta <: \alpha$, the new system specification $\gamma' <: \gamma$.

That is, the "components" of a system can be replaced with more specific implementations and the resulting system is still an implementation of the composite specification of the connected whole. This allows us to build systems abstractly using only specifications of the parts without considering their actual implementations that are assumed to exists.

Our approach resembles that of the *behavioral semantics* developed by Greif in [9], where she introduced a specification language for *actors* based on *causal axioms* with semantics as sets of completed computations. We use the system scenarios in $(pre\text{-})\mathcal{D}es(X)$ to provide the *domain of discourse* for a many-sorted first-order logic. We present here the core of a language called *Network Resource Calculus* that can be used to express pre-DES system specifications in a formal language.

The variables of the logic range over events E, pre-DESes O, natural numbers $\mathbb{N}$, and the points of a topological space X. The signature for the new non-logical predicates and functions is $(\leq: E \times E \to 2, \in: E \times O \to 2, \pi_1 : E \to X, \pi_2 : E \to \mathbb{N})$. Equality is supported for all variables and axioms of natural numbers and the space X are assumed for the reasoning about the event data. In our intended interpretation, each *sentence* of the logic without free variables is mapped to a statement about system scenario in $pre\text{-}\mathcal{D}es(X)$, describing the relations between the known interfaces of the system. A specification encodes laws that constraint the component pre-DES scenarios and how they are allowed to coincide. Overloaded symbol $\in$ is given the interpretation that if $\in (e, c)$, the event e is required to take place in a particular pre-DES c. The overloaded predicate $\leq$ maps to the causal structure attached to the underlying topological space. More specific kinds of topological spaces may define additional structure and axioms that can be used in specifications of systems in such spaces. The projection maps π_1 and π_2 for event data are interpreted in the expected way. As axioms, we assume *extensionality* for the pre-DESes, which means that if two pre-DESes share the same events they are equal. Similarly events are equal if their elements are the same and the DES condition in Definition 4 is formulated for each type of space used. Also the morphisms, or *connections* between the components, are expressed as sentences of the logic.

The syntax of NRC includes type signatures for networks

$$n : A_1 \wedge \cdots \wedge A_n \to C$$

where n is a name and A_i and C are sentences of the logic without free variables. A_i are assumptions needed for building a network n implementing the specification C. Basically, A_i are the specifications of the constituent subnetworks that are connected in n to a single diagram, formulas describing the constraints added by the new connections in n, and some general assumptions about the topological space X on which

---

[2] A set of scenarios instead of set of behaviors.

the pre-DESes are defined. We can define the *proof object* n for a diagram of type $A_1 \wedge \cdots \wedge A_n \to C$ with the syntax

$$n(n_1, \ldots, n_n) = d_1 \ldots d_k$$

where n_i are proof objects of A_i respectively. Some of the n_i may have been declared earlier without explicit proof, in which case they form the hypotheses that are assumed to be true, such as pre-DES building blocks whose implementations are assumed to exist, new connection constraints, or assumptions about the topological space X. The remaining $n_i : A_i$ are the provided constituent networks that each implement one specification A_i. d_j form a sequence of machine checkable derivations, where each d_j follows from the syntactic *rules of inference* applied to some earlier derivations $d_i, i \leq j$ or A_l and the last derivation $d_k = C$.

A connection c_i in n from $a_i : O$ to $b_i : O$ can be expressed as a parametrized formula with exactly two free variables of the type O and they are of the form $c_i(a_i, b_i)$, one for each morphism in the constructed diagram. Each c_i describes the condition which is true for the limit of the diagram n when the connection between the given two pre-DESes is included. Before a DEII can be connected, it has to be shown to be different from the already connected DEIIs. When we connect a DEOI o to a DEII i interface via medium DES $\mathcal{M}$ using identity projections, the new connection can be expressed

$$id(o, i) = \forall e : E\ e \in o \leftrightarrow e \in i$$

We were inspired by the Curry-Howard correspondence between proofs and programs in intuitionistic logic [10]. In NRC we embed the construction of a network from smaller *resources* inside classical logic and record the used assumptions by using the more restricted form for network types. A proof $n : A \to C$ means that if we have implementations of components and connections in type A, the resulting system implements specification C. Alternatively n can be considered a framework, that implements functionality C if components in A are provided. The specifications for networks can be used for arbitrary abstractions by deriving weaker consequences from the type denoting a larger set of system scenarios. The specifications can also be reformulated so that their subsentences can be interpreted as new abstracted networks with morphisms from and to their interface pre-DESes.

As a concrete example, a *pipe*, that forwards all events from DEII input port to DEOI output (and may output other events too) could be given the type

$$\exists i, o, p : O\ DEII(i) \wedge DEOI(o) \wedge SC(p, a) \wedge SC(p, b) \wedge$$
$$\forall e_1 : E e_1 \in i \to \exists e_2 : E(e_2 \in o \wedge e_1 \leq e_2 \wedge \pi_2(e_1) = \pi_2(e_2))$$

where $SC(x, y)$ encodes the condition of subcomponent morphisms from x to y. We could now prove that by connecting two pipes in sequence and hiding the implementation, we can produce a new pipe as shown in Fig. 4.

The reason for the restriction to functional networks here is that the general relational case is not compatible with the abstraction built into the logic. By separating inputs and outputs we can always make the antecedents stricter and relax consequents producing correct specifications saying less about the system. By assuming that each component must have at least one scenario including one for all possible combinations of input, the system has at least one global scenario.

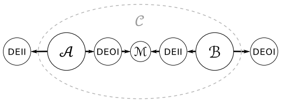

Fig. 4. Pipe DESes $\mathcal{A}$ and $\mathcal{B}$ connected in sequence form a single pipe $\mathcal{C}$.

A. Spacetime Charts

We also introduce a simple notation called *spacetime chart (STC)* for the visual representation of a subset of possible types expressible in NRC used for the description of functional event dependencies common in applications. In Fig. 5 we have given the corresponding STC for the external behavior of the pipe ES given earlier. While such charts are much easier to understand than the sentences of NRC, we can give them exact semantics as a mapping to NRC specifications.

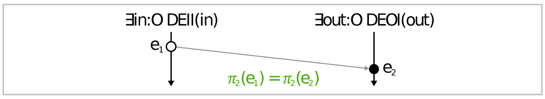

Fig. 5. Spacetime chart for the external behavior of a pipe.

In the general case, a STC consists of vertical arrows for pre-ESes, named events that are drawn as circles, and arrows between events representing causal order between events in such a way that if there is an arrow from e_1 to e_2, maps this to formula $e_1 \leq e_2$ and event e_1 is drawn higher than $e2$. If event e is drawn on top of pre-DES arrow o, translates this to formula $e \in o$. The diagram may also include NRC formulas as additional constraints such as the requirement $\pi_2(e_1) = \pi_2(e_2)$ in Fig. 5. The vertical arrows are labeled with either an existential or universal quantification for a pre-DES variable followed by a NRC sentence characterizing the pre-DES. Events that are drawn as unfilled circles are called antecedents and the remaining events are called consequents. The meaning of an STC can now be translated to a sentence

$$\exists o_f, \ldots, o_g : O \wedge chr_f(o_f) \wedge \cdots \wedge chr_g(o_g) \wedge$$
$$\forall o_{f'}, \ldots, o_{g'} : O \forall e_i, \ldots, e_{i'} : E\ chr_{f'}(o_{f'}) \wedge \cdots \wedge chr_{g'}(o_{g'}) \wedge$$
$$(e_k \in o_a) \wedge \cdots \wedge (e_l \in o_b) \wedge con_{q''1} \wedge \cdots \wedge con_{q''n} \to$$
$$\exists e_j, \ldots, e_{j'} : E(e_m \in o_c) \wedge \cdots \wedge (e_n \in o_d) \wedge con_{q'''1} \wedge \cdots \wedge con_{q'''n}$$

where antecedents and consequents are divided on the different sides of the implication and additional constraints including the causal relations con_s are included to antecedents iff they only refer to other antecedents. The conditions $chr_{s'}$ characterize the pre-DESes of the diagram. If an event is not drawn on any arrow, a universally quantified pre-DES is assumed.

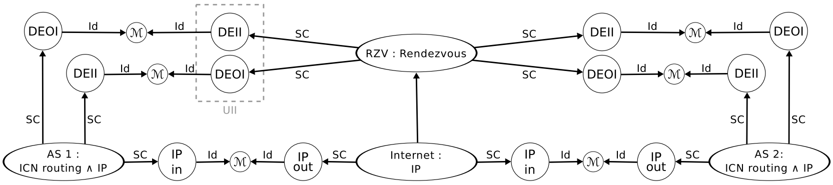

Fig. 6. A graphical depiction of an example diagram of ESes with identity maps and morphisms that can be expressed as compositions omitted. The diagram could be encoded as an NRC map that documents the different types of ESes and morphisms used. (SC = subcomponent morphism, Id = identity projection)

V. Use Case: Protocol Specification in ICN

An *Information-centric network (ICN)* can be understood in the widest sense as a system that facilitates *sharing of data* [8]. Without additional restrictions, such as *scopes* [8], on who can receive the data, this leads to flexibility in the extension of the functionality as anybody can intercept data and *publish* new data based on it. For example, a network management system could read the network descriptions used by an *information-centric* (IC) routing protocol. On the downside, this freedom leads to a monolithic architecture and the system will adapt badly to changes in the underlying assumptions of the design. This leads to our idea that ICN equivalent of a protocol is an *information structure (IS)*, which is a specification of a set of data types and *causal axioms* that describe how the shared state of the system evolves *globally* when values of the types under question are published. We specify this abstract notion of shared state, *an information space*, to be a DES with variable number of DEOI and DEII interfaces. Publishing a data item d means that the information space $\mathcal{S}$ receives an event $e, \pi_2(e) = d$ in one of its DEIIs. $\mathcal{S}$ implements IS x if the causal axioms of x follow from the specification of $\mathcal{S}$.

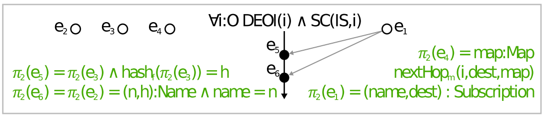
e_2
e_3
e_4
∀i:O DEOI(i) ∧ SC(IS,i)
e_1
e_5
e_6
$\pi_2(e_4) = map{:}Map$
$\pi_2(e_5) = \pi_2(e_3) \wedge hash_f(\pi_2(e_3)) = h$
$nextHop_m(i, dest, map)$
$\pi_2(e_6) = \pi_2(e_2) = (n,h){:}Name \wedge name = n$
$\pi_2(e_1) = (name, dest) : Subscription$

Fig. 7. The causal axiom of name-based subscription IS as a spacetime chart.

As a toy example[3] we show in Fig. 7 the causal axiom of *name-based subscription* IS, which introduces data types $Subscription$, $Name$, and Map. This specification states that when the information space $\mathcal{S}$ is given the data $\pi_2(e_1)$ encoding a value of data type $Subscription$, a pair $(name, dest)$, the data $\pi_2(e_3)$ and its naming information $\pi_2(e_2)$ will be forwarded with events e_5 and e_6 to all interfaces i that are connected to a next hop towards destination $dest$ according to some routing metric m in any map published in $\mathcal{S}$. The values of data type $Name$ are pairs (n, h), which associate a name n with the hash h that identifies the data item $\pi_2(e_3)$.

The above example leads immediately to another use case for NRC as the partial network maps used for routing could be represented as NRC descriptions. As an example, Fig. 6 shows a graphical depiction of a diagram that specifies an abstract map of connections between two ASes and a *rendezvous service* shared by them. The arrow from the ES representing the rest of the Internet to the rendezvous ES stands for an implementation dependency between them. Such information may be important for the description of overlay networks or multipath routing, where independent paths are preferred for better resilience to network failures.

Compared to some *network description languages* such as NDL [11] and Nimrod maps [12], NRC can express variety of dependencies between components, changing network topologies in spacetime, and use flexible abstraction for the description with exact compositional semantics. Also, when the application requires a network that has some property known at compile-time, NRC allows the network descriptions to carry a machine-readable proofs that the given network implements the required specification.

References

[1] G. Winskel and M. Nielsen, *Handbook of Logic in Computer Science - Volume 4: Semantic Modelling*. Oxford University Press, 1995, ch. Models for concurrency, pp. 1–148.
[2] B. C. Pierce, *Basic Category Theory for Computer Scientists*. The MIT Press, 1991.
[3] J. A. Goguen, “Sheaf Semantics for Concurrent Interacting Objects,” *Mathematical Structures in Computer Science*, vol. 2, no. 02, pp. 159–191, 1992.
[4] S. Mac Lane and I. Moerdijk, *Sheaves in Geometry and Logic: A First Introduction to Topos Theory*. Springer-Verlag, 1992.
[5] J. D. Brock, “A Formal Model of Non-determinate Dataflow Computation,” Ph.D. dissertation, Massachusetts Institute of Technology, 1983.
[6] C. Hewitt and H. Baker, “Laws for Communicating Parallel Processes,” in *IFIP’77*, 1977.
[7] S. Mac Lane, *Categories for the Working Mathematician*. Springer, 1971.
[8] K. Visala, D. Lagutin, and S. Tarkoma, *The Future Internet - Future Internet Assembly 2013: Validated Results and New Horizons*, ser. b. Springer, 2013, vol. 7858, ch. Towards a Minimal Core for Information-Centric Networking, pp. 39–51.
[9] I. G. Greif, “SEMANTICS OF COMMUNICATING PARALLEL PROCESSES,” Ph.D. dissertation, Massachusetts Institute of Technology, September 1975.
[10] M. H. Sørensen and P. Urzyczyn, *Lectures on the Curry-Howard Isomorphism*. Elsevier, 2006.
[11] J. J. v. d. Ham, F. Dijkstra, F. Travostino, H. M. Andree, and C. T. d. Laat, “Using RDF to Describe Networks,” *Future Generation Computer Systems*, vol. 22, no. 8, pp. 862–867, 2006.
[12] I. Castineyra, N. Chiappa, and M. Steenstrup, “RFC1992: The Nimrod Routing Architecture,” IETF, Tech. Rep., 1996.

---

[3] This specification does not allow failures, assumes an infinite storage for the information space and does not implement any security features.

Aalto-ST 10/2014

ISBN 978-952-60-5794-1 (pdf)
ISSN-L 1799-4896
ISSN 1799-4896
ISSN 1799-490X (pdf)

Aalto University
School of Science
Department of Computer Science and Engineering
www.aalto.fi

BUSINESS +
ECONOMY

ART +
DESIGN +
ARCHITECTURE

SCIENCE +
TECHNOLOGY

CROSSOVER

DOCTORAL
DISSERTATIONS