Dynamic Reconfiguration via Typed Modalities

Tutu, Ionut; Chirita, Claudia; Fiadeiro, José Luiz

Published in:
Formal Methods

DOI:
10.1007/978-3-030-90870-6_32

Publication date:
2021

Document Version
Peer reviewed version

Author Accepted Manuscript: Tutu, I, Chirita, C & Fiadeiro, JL 2021, Dynamic Reconfiguration via Typed Modalities. in M Huisman, C Păsăreanu & N Zhan (eds), Formal Methods: 24th International Symposium, FM 2021, Virtual Event, November 20–26, 2021, Proceedings. 1 edn, Lecture Notes in Computer Science , vol. 13047, Springer , pp. 599-615. https://doi.org/10.1007/978-3-030-90870-6_32

Dynamic Reconfiguration via Typed Modalities

Ionuţ Ţuţu[1,3], Claudia Elena Chiriţă[2], and José Luiz Fiadeiro[3]

[1] Simion Stoilow Institute of Mathematics of the Romanian Academy, Romania
ittutu@gmail.com
[2] School of Informatics, University of Edinburgh, UK
claudia.elena.chirita@gmail.com
[3] School of Science and Engineering, University of Dundee, UK
jfiadeiro@dundee.ac.uk

Abstract. Modern software systems are increasingly exhibiting dynamic-reconfiguration features analogous to naturally occurring phenomena where the architecture of a complex changes dynamically, at run time, on account of interactions between its components. This has led to a renewed interest in modal logics for formal system development, building on the intuitive idea that system configurations can be regarded as local models of a Kripke structure, while reconfigurations are captured by accessibility relations. We contribute to this line of research by advancing a modal logic with varying quantification domains that employs typed modalities and dedicated modal operators to specify and reason about a new generation of Kripke structures, called dynamic networks of interactions, that account for the context of a system's dynamics, identifying which actants have triggered a reconfiguration and what are its outcomes. To illustrate the expressiveness of the formalism, we provide a specification of the biological process of membrane budding, which we then analyse using a sound and complete proof-by-translation method that links dynamic networks of interactions with partial first-order logic.

Keywords: Reconfigurable systems · Modal logic · Typed modalities · Standard translation · Partial first-order logic

1 Introduction

The process of dynamic reconfiguration is central to understanding the way modern software systems operate. For instance, in service-oriented computing, where software applications need to evolve in response to changing requirements or environmental conditions, reconfigurability accounts for run-time architectural transformations [12]; therefore, the process is closely related to self-adaptability – a key feature of the systems operating today in cyberspace. Dynamic reconfigurability is also one of the main processes that drive the functioning of cyber-physical systems. In actor-network theory – a modelling framework for cyber-physical-system protocols that we have recently formalized in [13] in connection with

Robin Milner's bigraphs [22] – the topology of a network is modified dynamically, during its execution, as a result of interactions taking place between actors.

The topic is duly receiving growing attention in the formal-methods literature, notably through new mathematical models as well as specification and reasoning tools based on modal and hybrid(ized) logics that explore the intrinsic connection between reconfigurability and Kripke structures (e.g., [21,18,8,10]). In their most basic form, hybrid logics enrich ordinary modal logics with nominals – designating states in Kripke structures – and a local-satisfaction operator that enables a change of perspective from the state under consideration to another state that is named (e.g., [1,4]). Our work on actor-network theory [13,30] highlighted two important limitations of the hybrid-logic approach: it offers no support for tracking, which means that even though reconfigurations are triggered by specific interactions between network components, there is no way to pin down the components involved in an interaction or to track them through the reconfiguration; moreover, the systems thus modelled are closed, meaning that reconfigurations can alter the locality or inter-connectivity of network components but they cannot add or remove components to or from a given configuration. This has a significant impact on the specification capabilities of the approach.

In this work, we set out to address those limitations by introducing a modal logic for reconfigurable systems with two new features: (*a*) it makes use of typed modalities where inner and outer types indicate which network components have triggered a reconfiguration and what the outcomes are; and (*b*) it allows for varying quantification domains – in lieu of the more conventional constant domains – thus enabling the modelling of open systems. The first feature, in particular, is a notable departure from ordinary modal logics because the accessibility relations of Kripke structures, whilst still capturing reconfigurations, no longer link network configurations directly; instead, they define inter-network connections between different states or instances of the components involved in reconfigurations.

The paper is structured as follows. In Section 2, we highlight some of the challenges raised by dynamic reconfigurations on conventional modal-logic frameworks; for that purpose, we present a case study based on the biological process of membrane budding [19], which is paradigmatic for open systems. In Section 3, we introduce a new kind of Kripke structures – called dynamic networks of interactions – that underlie the modal-logic formalism proposed in this paper, which we then apply to the case study. Finally, in Section 4, we outline a sound and complete proof-by-translation technique for dynamic networks of interactions and we put it to use in order to analyse the membrane-budding process.

2 Membrane budding

Dynamic networks of interactions can be broadly described as networks whose configurations change dynamically, at run time, as a result of interactions taking place among their components. Where examples abound in computer-based or

software-enabled systems such as cyber-physical systems, to help us develop an intuitive understanding of the nature of such reconfigurations we look instead at membrane budding – a biological process that is emblematic of eukaryotic cells through which protein and lipid molecules are transported between organelles such as the endoplasmic reticulum and the Golgi complex. Biological processes are being used, more and more, for developing new modes of computation (e.g., [20,25]), and mathematical modelling techniques play an important part in reasoning about such processes; our running example serves both purposes.

In a nutshell, membrane budding facilitates the transport of cargo molecules along intracellular pathways by enclosing them in carrier vesicles, which in turn develop during the budding process through the deformation of the donor organelle's membrane. As a reference, we consider the simplified model of membrane budding from [19]. The biological model explains (*a*) the assembly of protein coats on the membrane of a source organelle, (*b*) the formation of a bud as a result of interactions between coat proteins and the membrane, (*c*) the detachment of the bud from the organelle, hence forming a vesicle, (*d*) the uncoating of the vesicle, and finally (*e*) the fusion with the membrane of the target organelle.

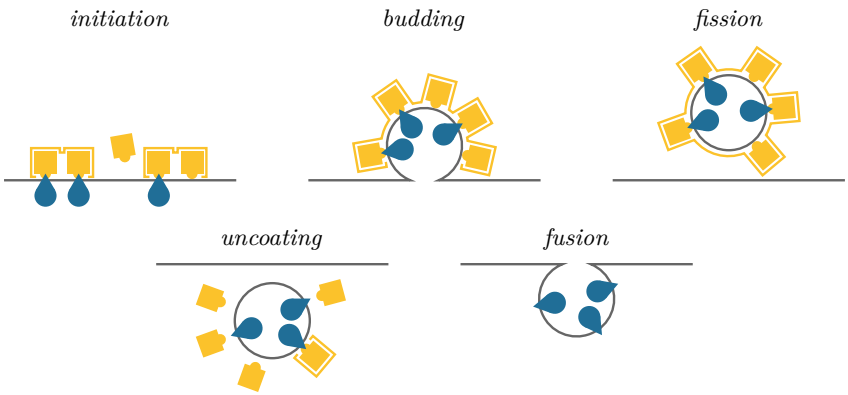

Fig. 1. The five stages of the membrane-budding process – black lines represent membranes of organelles, buds, or vesicles; yellow squares represent proteins; blue drops represent cargo molecules; and yellow contours around proteins represent coats

The entire process is depicted in Figure 1, and unfolds as follows. The first stage – called *initiation* (cf. [19, Table 2]) – begins with a set of reactions that lead to the recruitment of specific proteins (represented in our figure as yellow squares); those proteins eventually bind to the donor membrane (represented as the lower black line), cluster into coats, and capture cargo molecules (represented as blue drops) from the underlying membrane. The second stage – *coat propagation*, or *budding* – corresponds to a deformation of the membrane under the effect of the coat assembly. The third stage – *vesicle formation*, or *fission* – is triggered by

the completion of the coat, which depends on the kinds of proteins involved, and consists in the detachment of the bud from the membrane by scission. The last two stages – *uncoating* and *fusion* – occur at the arrival of the vesicle in the proximity of the target organelle. The vesicle then loses its coat and fuses with the membrane, thus completing the delivery of its cargo molecules.

Using modalities to capture reconfigurations. Dynamic networks of interactions similar to the ones arising in membrane budding are essentially discrete reconfigurable systems; hence, it is natural to formalize them as Kripke structures whose states/worlds correspond to network configurations (specific arrangements of organelles, proteins, cargo molecules, etc., at a given time) and whose transitions capture reconfigurations (like those occurring during budding or fission). This opens the possibility to use standard modal-logic formalisms as in [7,21,11,18,30], among other, to specify and reason about reconfigurations.

The modal-logic approach to dynamic reconfigurations ordinarily relies on the use of logical systems that blend features of a base logic (in our case, of many-sorted first-order logic with equality) with features that are conventionally attributed to modal logic such as possible-worlds semantics and the use of modal operators. We use base-logic sentences to specify relationships between network components in a given configuration, and modal-logic operators to describe the effects of reconfigurations. To illustrate the modus operandi, consider the binding of proteins to organelles during the initiation stage of membrane budding. Whenever a free (i.e., unbound) protein p is close enough to an organelle o, binding p to o results in a new configuration where p is part of a coat c formed on the membrane of o. Formally, we could write:

$$\mathsf{close\text{-}enough}(p, o) \wedge \mathsf{free}(p) \Rightarrow [\mathsf{bind}]\exists c \cdot \mathsf{part\text{-}of}(p, c) \wedge \mathsf{brane}(c) = \mathsf{brane}(o)$$

where close-enough, free, part-of, and brane are relation/operation symbols from a base first-order signature (suitably axiomatized, as we discuss in Section 3), and bind is a modality. This sentence is a typical example of the use of modal logic to formalize pre/post-conditions in program specification and verification [24]: the consequent $\exists c \cdot \mathsf{part\text{-}of}(p, c) \wedge \mathsf{brane}(c) = \mathsf{brane}(o)$ captures the effect of any reconfiguration performed by the action (modality) bind on any configuration that satisfies the antecedent $\mathsf{close\text{-}enough}(p, o) \wedge \mathsf{free}(p)$.

However, this usage of modalities is not adequate for our purposes because it fails to capture all bind-reconfigurations of interest whilst allowing other – biologically inconsistent – reconfigurations to occur. That is because the ramifications of the above axiomatization are global: under the usual semantics of the modal-logic operator [bind], given that variables p and o are implicitly universally quantified, bind-reconfigurations simultaneously affect *all* organelles o (in any configuration) and *all* unbound proteins p that are sufficiently close to o. Therefore, proteins that are sufficiently close to several organelles will necessarily end up being bound to all of them in the same resulting configuration – which is discordant with the biological model of membrane budding that we intend to formalize.

Another drawback of conventional modal-logic approaches to reconfigurable systems is that they make use of constant quantification domains (were variables such as p, o, and c in the sentence above are instantiated). That is, conventionally, reconfigurations cannot create or remove network components; their effects are limited to changing the states of components or the way they relate to other components. In regard to membrane budding, this constraint is admissible for proteins, whose states may change from free to bound, or vice versa; however, it is not admissible for coats, which are created during initiation and removed during the uncoating stage – nor for buds or vesicles, which have a similar transitory nature. Returning to the sentence above, the consequent variable c ought to refer to a newly generated coat that could not have existed in the configuration where p and o are instantiated (because p is unbound in that configuration).

Towards typed modalities and varying domains. In light of the discussion above, the next section brings forward a new modal-logic formalism that is specifically adapted to support the development/specification and analysis of dynamic networks of interactions. There are two major attributes that differentiate the formalism we propose from previous modal-logic approaches:

1. It makes use of typed modalities and generalized modal operators that account for the context in which reconfigurations take place.
2. It features varying quantification domains in order to accommodate open systems where components are created or removed on the fly as needed.

The second attribute belongs to an area of modal logic that has received considerable attention (e.g., [14,16,2,4]) mainly through exploring specific choices of kinds of first-order symbols and of Kripke structures; for instance, in [14], every individual (network component) needs to exist in some local domain, while [4] admits only constant-operation symbols and flexible predicates. Our contribution is in facilitating the full, unhindered use of many-sorted first-order logic in conjunction with modal operators; this can be seen as a varying-domain counterpart of the work reported in [11], which employs shared quantification domains.

Typed modalities are, to the best of our knowledge, a new development. They provide a mechanism that supports tracking – reminiscent of [22, Chapter 11] – so that one can identify, for every reconfiguration, which actants (network components) have triggered it and what are its specific outcomes. For instance, the bind-reconfigurations that we have considered thus far are triggered by interactions between proteins p and organelles o, and their outcomes are specific coats c that arise once p is attached to the membrane of o. Therefore, we can regard bind as a modality that is typed by proteins, organelles, and coats.

3 Logical support for dynamic networks of interactions

In order to address the concerns raised in Section 2, we propose a dedicated logical system, hereafter referred to as DNI, that supports the formal development of

dynamic networks of interactions. We present the logic in familiar fashion, focusing on *signatures* (organized collections of non-logical symbols), *models* (Kripke structures over which the symbols declared in signatures are given semantics), *sentences* (built from symbols declared in signatures), and *satisfaction relations* (establishing which sentences hold at given configurations of a network).

First-order-logic preliminaries. DNI combines first-order quantifiers with features specific to modal and, in particular, hybrid logics [1]. To establish the terminology and notations used throughout the paper, we recall a few basic notions pertaining to first-order logic in an algebraic setting – see [17,11] for more details.

By *first-order signature* we mean a triple $\Sigma = \langle S, F, P\rangle$, where S is a set of *sorts* (which we typically denote by s), F is an $S^* \times S$-indexed family of sets of *operation symbols* $\sigma\colon u \to s$ (named σ, having *arity* u and *sort* s), and P is an S^*-indexed family of sets of *predicate symbols* $\pi\colon u$ (named π, of *arity* u). An $\langle S, F, P\rangle$-*model* is a structure M that interprets: every sort s in S as a set M_s, called the *carrier set* of s in M; every finite sequence of sorts $u = s_1 \cdots s_n$ as the Cartesian product $M_{s_1} \times \cdots \times M_{s_n}$; every operation symbol $\sigma\colon u \to s$ in F as a function $M_\sigma\colon M_u \to M_s$; and every predicate symbol $\pi\colon u$ in P as a subset $M_\pi \subseteq M_u$. In relation to the syntax of first-order logic, we consider *terms* built, inductively, from operation symbols, including constants. We use terms to form *equational atoms* $t_1 = t_2$, where t_1 and t_2 are terms of the same sort, and *relational atoms* $\pi(t)$, where π is a predicate symbol and t is a *tuple of terms* corresponding to the arity of π. Full first-order sentences are obtained from equational and relational atoms using Boolean connectives and quantifiers (over variables which we treat as new constant-operation symbols).

The presentation of DNI makes use of first-order signature extensions, model reducts, and closed submodels. We say that a signature Σ' *extends* Σ, and we write $\Sigma \subseteq \Sigma'$ when every sort and every operation/predicate symbol in Σ belongs to Σ' as well. In particular, for every set X of variables (distinct from the constant-operation symbols in Σ), we obtain the extension $\Sigma(X)$ by adding the variables in X to Σ as new constant-operation symbols. For every Σ'-model M' we define the *reduct* $M'{\restriction_\Sigma}$ as the Σ-model M obtained from M' by 'forgetting' the interpretation of all symbols in Σ' that do not belong to Σ – and in that case we also say that M' is a Σ'-*expansion* of M. Lastly, we say that M is a *closed submodel* of D, and we write $M \subseteq D$, when $M_s \subseteq D_s$ for every sort s in Σ, $M_\sigma(a) = D_\sigma(a)$ for every operation symbol $\sigma\colon u \to s$ and every tuple of arguments $a \in M_u$, and $M_\pi = D_\pi \cap M_u$ for every predicate symbol $\pi\colon u$.

DNI-signatures. Depending on where and how they are meant to be interpreted, we distinguish between (a) symbols that are meant to be interpreted only globally, at network level, irrespective of configurations, (b) symbols that are evaluated only locally, at every configuration of a network, yielding interpretations that may vary from one configuration to another, and (c) symbols that are evaluated both globally and locally, in which case we require their interpretations to agree. In

keeping with standard modal-logic terminology, we refer to the symbols belonging to the first and third category as *rigid*, and to those in the second category as *flexible*. In addition, we refer to symbols that belong to the third category as *shared* to indicate that their interpretation is jointly defined across configurations.

Besides first-order symbols, DNI-signatures make use of *nominals* – as in hybrid logic [1] – through which specific configurations can be designated and of *typed modalities* with *inner* and *outer types* that match the triggers and outcomes of their corresponding reconfigurations.

Formally, a *DNI-signature* is a tuple $\Xi = \langle \Delta, \Omega, N, \Lambda \rangle$ consisting of:

- first-order signatures Δ and Ω over the same set S of sorts; Δ comprises the global, and thus *rigid*, symbols of Ξ, while Ω comprises the local symbols of Ξ; the symbols in $\Sigma = \Delta \cap \Omega$ are *shared*, and those in $\Omega \setminus \Sigma$ are *flexible*;
- a set N of *nominals*, or *configuration designators*; and
- an $S^* \times S^*$-indexed family Λ of sets of *modalities*; given two sequences of sorts $u, v \in S^*$, we typically write $\lambda \colon u \nrightarrow v$ for $\lambda \in \Lambda_{u,v}$ and we say that u is the *inner type* and v is the *outer type* of λ.

Example 1. For membrane budding, we define a DNI-signature with seven sorts:

> **sorts** Organelle, Brane, Protein, Cargo, Coat, Bud, Vesicle

These correspond, in order, to organelles, membranes (of organelles, but also of buds or vesicles), proteins, cargo molecules, coats, buds, and vesicles.

Intuitively, coats are clusters of proteins bound to a given membrane; as such, we define two rigid operations: just, which is used to form simple coats out of single, isolated proteins, and add, which is used to enlarge existing coats.

> **rigid ops** just : Protein → Coat
> add : Protein Coat → Coat

These operations are rigid but not shared because, when applied to proteins and coats in a configuration w, they may evaluate to coats that do not belong to w. For instance, if a protein p is unbound in w, then the coat $\mathsf{just}(p)$ cannot belong to w – because that would imply that p is bound to a membrane.

We also consider two membership predicates: part-of to determine whether a protein belongs to a coat, and enclosed to determine whether a cargo molecule is surrounded by a given membrane. The first one is shared because coats are intended to be uniquely determined by the proteins of which they are formed (independently of the configuration where said proteins and coats are located), while the second is flexible because the cargo molecules enclosed in a membrane may vary from one configuration to another as a result of their transport.

> **shared pred** part-of : Protein Coat
> **flexible pred** enclosed : Cargo Brane

Some entities have corresponding membranes. This is obvious for organelles, buds, and vesicles, but it also holds for coats, which are necessarily bound to specific membranes. We capture these attributes by overloading the operation

symbol brane. For the first three cases, we declare brane as a shared symbol because the membranes of organelles, buds, and vesicles are fixed and defined at every configuration; for the latter case, brane is flexible because the membrane of a coat varies depending on the organelle on which the coat is located.

shared op brane: $s \to$ Brane, where $s \in \{$Organelle, Bud, Vesicle$\}$
flexible op brane: Coat $\to$ Brane

We use an additional predicate to indicate if a protein is close enough to an organelle in order to be bound. This predicate is flexible because the position of a protein relative to an organelle may change from one configuration to another. For example, a protein that is attached (and thus close enough) to an organelle during the initiation stage will no longer be so during fission.

flexible pred close-enough: Protein Organelle

Similar predicates are used to indicate if a cargo molecule is close enough to a protein (in order to be captured) or if two proteins belonging to different coats are close enough to enable the two coats to merge into a bigger cluster.

The modal part of the signature consists of seven modalities: bind for binding a protein to an organelle, thus producing a coat; merge for merging two neighbouring coats into a bigger coat; capture, thorough which a cargo molecule is captured by a protein (bound to the membrane enclosing that molecule); etc.

modalities bind: Protein Organelle $\times$ Coat; merge: Coat Coat $\times$ Coat;
capture: Cargo Protein $\times \varepsilon$; bud: Coat $\times$ Bud;
split: Bud $\times$ Vesicle; uncoat: Protein Vesicle $\times \varepsilon$;
fuse: Vesicle Organelle $\times \varepsilon$.

Some of the modalities listed above, such as capture and uncoat, have empty outer types, which we indicate by ε; this means that their corresponding reconfigurations have no specific outcomes other than the (implicit) change of configuration.

Dynamic networks. A *Ξ-model*, or *network*, $\langle D, W, M\rangle$ consists of:

- a first-order Δ-model D, called the *global domain* of the network;
- a *Kripke frame* W; we denote by $|W|$ the set of *possible worlds* or *configurations* of W, and by $W_i \in |W|$ and $W_\lambda \subseteq |W| \times |W|$ the interpretations of nominals $i \in N$ and modalities λ in Λ (regardless of their inner/outer types);
- for every $w \in |W|$, a *local* Ω-model M_w such that $M_w{\restriction_\Sigma} \subseteq D{\restriction_\Sigma}$;[4]
- for every $(w, w') \in W_\lambda$ with $\lambda\colon u \times v$, a relation $M_{w\lambda w'} \subseteq M_{w,u} \times M_{w',v}$ between elements in M_w corresponding to the sorts in u (triggering actants for λ) and elements in $M_{w'}$ corresponding to the sorts in v (outcomes of λ).

Signature extensions and network reducts. The formalism we propose makes use of two kinds of variables: rigid first-order variables for referencing network components or data, and nominal variables (inherited from ordinary hybrid logic)

[4] Recall that, in the context of DNI, we denote by Σ the intersection $\Delta \cap \Omega$.

for referencing configurations. Both kinds of symbols need to be distinct from the constants and nominals declared in the signature under consideration.

For every DNI-signature $\Xi = \langle \Delta, \Omega, N, \Lambda \rangle$ over a set S of sorts, and every S-sorted set X of first-order variables and set K of nominal variables, we obtain an extended signature $\Xi(X;K) = \langle \Delta(X), \Omega, N \uplus K, \Lambda \rangle$ and thus the inclusion $\Xi \subseteq \Xi(X;K)$ by adding the first-order variables in X to Δ as new rigid constants and the nominal variables in K to N as new nominals. If $X = \emptyset$, we denote the extended signature by $\Xi(K)$; similarly, if $K = \emptyset$, we use the notation $\Xi(X)$.

Similarly to first-order logic, every $\Xi(X;K)$-network $\langle D', W', M' \rangle$ can be reduced to a Ξ-network $\langle D, W, M \rangle = \langle D', W', M' \rangle\restriction_{\Xi}$ simply by forgetting the interpretation of the newly added symbols from X and K. That is, $\langle D, W, M \rangle$ is a network with the same carrier sets and possible worlds as $\langle D', W', M' \rangle$, and which interprets all first-order and modal symbols in Ξ just as $\langle D', W', M' \rangle$.

Sentences. The terms over a DNI-signature Ξ are built by freely putting together operation symbols from the two first-order signatures that underlie it. To that end, we define *DNI-terms* over Ξ as first-order terms over the compound signature $\Delta \cup \Omega$. For every sort s in Ξ, we denote by $T_{\Xi,s}$ the set of Ξ-terms of sort s, and for every sequence of sorts $u = s_1 \cdots s_n$, we denote by $T_{\Xi,u}$ the Cartesian product $T_{\Xi,s_1} \times \cdots \times T_{\Xi,s_n}$. We further inherit from first-order logic both *equational atoms* $t_1 = t_2$, where t_1 and t_2 are Ξ-terms of the same sort, and *relational atoms* $\pi(t)$, where $\pi \colon u$ is a relation symbol (with arity u) in $\Delta \cup \Omega$ and $t \in T_{\Xi,u}$ is a tuple of Ξ-terms corresponding to the sorts in u.

Core *DNI-sentences* over Ξ are defined according to the following grammar:

$$\rho ::= e \mid \flat l \mid i \mid \neg\rho \mid \rho \wedge \rho \mid @i \cdot \rho \mid [t\, \lambda\, z]\rho' \mid \forall X; K \cdot \rho^{\dagger}$$

where (a) e is either an equational or a relational atom over Ξ; (b) $\flat$ is a *local-membership* operator and l is a Ξ-term; (c) i is a nominal, which, together with e and $\flat l$, is also considered an atomic sentence; (d) $@$ is a special *local-satisfaction* operator – as in hybrid logics; (e) $\lambda \colon u \rightarrowtail v$ is a modality, t is a tuple of terms corresponding to the sorts in u, z is a tuple of first-order variables corresponding to the sorts in v, and ρ' is a sentence over the extended signature $\Xi(z)$; and lastly, (f) X is a finite set of first-order variables, K is a finite set of nominal variables, and $\rho^{\dagger}$ is a sentence over the extended signature $\Xi(X;K)$. When either X or K is empty, we denote $\forall X; K \cdot \rho^{\dagger}$ by $\forall K \cdot \rho^{\dagger}$ or $\forall X \cdot \rho^{\dagger}$, respectively.

For convenience, other logical connectives can be defined in terms of the basic ones in the usual manner. We denote disjunctions by $\rho_1 \vee \rho_2$, implications by $\rho_1 \Rightarrow \rho_2$, equivalences by $\rho_1 \Leftrightarrow \rho_2$, possibility statements by $\langle t\, \lambda\, z \rangle \rho$, and existentially quantified sentences by $\exists X; K \cdot \rho$ – together with the simplified forms $\exists X \cdot \rho$ and $\exists K \cdot \rho$, as described above. In addition, we consider *locally quantified sentences* $\forall \flat X \cdot \rho$ and $\exists \flat X \cdot \rho$, where $\forall \flat X \cdot \rho$ stands for $\forall X \cdot \bigwedge\{\flat x \mid x \in X\} \Rightarrow \rho$ and $\exists \flat X \cdot \rho$ stands for $\exists X \cdot \bigwedge\{\flat x \mid x \in X\} \wedge \rho$ – this matches the usual notion of *actualist quantification* from first-order modal logic [14]. Lastly, the *store* operator $\downarrow k \cdot \rho$ used in hybrid logics can be defined as $\forall \{k\} \cdot k \Rightarrow \rho$.

The satisfaction relation. As with any modal logic, we distinguish between local – i.e., at configuration level – and global satisfaction of sentences by models (networks); the former takes precedence. To define satisfaction, we need to take a closer look at the way operation and relation symbols of a DNI-signature Ξ are interpreted locally, at given configurations of a network $\langle D, W, M\rangle$.

Interpretation of terms. For every configuration $w \in |W|$, every rigid operation $\sigma\colon u \to s$ is interpreted as a function $\langle D, W, M\rangle^w_\sigma\colon D_u \to D_s$ defined by $\langle D, W, M\rangle^w_\sigma(a) = D_\sigma(a)$ for all $a \in D_u$. In contrast, every flexible operation $\sigma\colon u \to s$ is interpreted as a partial function $\langle D, W, M\rangle^w_\sigma\colon D_u \nrightarrow D_s$, where $\langle D, W, M\rangle^w_\sigma(a)$ is defined and equal to $M_{w,\sigma}(a)$ if and only if $a \in M_{w,u}$.

The interpretation of operation symbols generalizes to DNI-terms in a straightforward inductive way: given an operation symbol $\sigma\colon u \to s$ and a tuple of terms $t \in T_{\Xi,u}$, the interpretation of the compound term $\sigma(t)$ at a configuration w, denoted $\langle D, W, M\rangle^w_{\sigma(t)}$, is defined and equal to $\langle D, W, M\rangle^w_\sigma(\langle D, W, M\rangle^w_t)$ if and only if the interpretation of t is defined at w and it belongs to the domain of $\langle D, W, M\rangle^w_\sigma$. Whenever we write $\langle D, W, M\rangle^w_t$, we implicitly assume that (the interpretation of) t is defined at w in the network $\langle D, W, M\rangle$.

Interpretation of relation symbols. Interpreting relation symbols is significantly simpler: if $\pi\colon u$ is a rigid relation symbol, we define $\langle D, W, M\rangle^w_\pi$ as D_π; otherwise (i.e., when $\pi\colon u$ is flexible), we define it as $M_{w,\pi} \subseteq D_u$.

The *local-satisfaction relations* of DNI are defined inductively, as follows:

- $\langle D, W, M\rangle \models^w_\Xi t_1 = t_2$ if $\langle D, W, M\rangle^w_{t_1} = \langle D, W, M\rangle^w_{t_2}$, meaning that both terms are defined at w, and their interpretations are equal;
- $\langle D, W, M\rangle \models^w_\Xi \pi(t)$ if $\langle D, W, M\rangle^w_t \in \langle D, W, M\rangle^w_\pi$;
- $\langle D, W, M\rangle \models^w_\Xi \flat l$ if $\langle D, W, M\rangle^w_l \in M_{w,s}$, when l is a Ξ-term of sort s;
- $\langle D, W, M\rangle \models^w_\Xi i$ if $w = W_i$, when i is a nominal;
- $\langle D, W, M\rangle \models^w_\Xi \neg\rho$ if $\langle D, W, M\rangle \not\models^w_\Xi \rho$;
- $\langle D, W, M\rangle \models^w_\Xi \rho_1 \wedge \rho_2$ if $\langle D, W, M\rangle \models^w_\Xi \rho_1$ and $\langle D, W, M\rangle \models^w_\Xi \rho_2$;
- $\langle D, W, M\rangle \models^w_\Xi @i \cdot \rho$ if $\langle D, W, M\rangle \models^{W_i}_\Xi \rho$;
- $\langle D, W, M\rangle \models^w_\Xi [t\,\lambda\,z]\rho'$ if $\langle D', W, M\rangle \models^{w'}_{\Xi(z)} \rho'$ for all transitions $(w, w') \in W_\lambda$ and all expansions $\langle D', W, M\rangle$ of $\langle D, W, M\rangle$ along the signature inclusion $\Xi \subseteq \Xi(z)$ such that $(\langle D, W, M\rangle^w_t, \langle D', W, M\rangle^{w'}_z) \in M_{w\lambda w'}$;
- $\langle D, W, M\rangle \models^w_\Xi \forall X; K \cdot \rho^\dagger$ if $\langle D^\dagger, W^\dagger, M^\dagger\rangle \models^w_{\Xi(X;K)} \rho^\dagger$ for all expansions $\langle D^\dagger, W^\dagger, M^\dagger\rangle$ of $\langle D, W, M\rangle$ along the signature inclusion $\Xi \subseteq \Xi(X; K)$.

Global satisfaction equates to local satisfaction at all possible worlds. More precisely, a Ξ-network $\langle D, W, M\rangle$ *globally satisfies* a DNI-sentence ρ, which we denote by $\langle D, W, M\rangle \models_\Xi \rho$, when $\langle D, W, M\rangle \models^w_\Xi \rho$ for all $w \in |W|$.

Example 2. We are now ready to outline the DNI-specification of membrane budding, which consists in a set E_{MB} of sentences written over the signature Ξ_{MB} introduced in Example 1. As usual, the semantics of the specification is given by the class of all Ξ_{MB}-networks that globally satisfy all sentences in E_{MB}.

We organize the specification in two categories of sentences:

1. Static constraints, which are meant to ensure that the configurations we specify are coherent with the biological model of membrane budding. Here, we include, for example, the relationship between proteins and coats, which is partly described in sentences (1) and (2) below, and what it means for a protein to be free, according to sentence (3). To keep the notations short within formulae, we write variables without their corresponding sorts; we let c range over coats, p and q range over proteins, and o over organelles.

$$\forall\{c\} \cdot \exists\{p\} \cdot \mathsf{part\text{-}of}(p, c) \tag{1}$$

$$\forall\{p, q\} \cdot \mathsf{part\text{-}of}(p, \mathsf{just}(q)) \Leftrightarrow p = q \tag{2}$$

$$\forall\flat\{p\} \cdot \mathsf{free}(p) \Leftrightarrow \neg\exists\flat\{c\} \cdot \mathsf{part\text{-}of}(p, c) \tag{3}$$

2. Sentences dealing with the dynamic aspects of membrane budding. In this case, for every kind of reconfiguration, we present both *progress axioms*, indicating under which conditions a reconfiguration may occur, and *effect axioms*, which describe the changes induced by that particular type of reconfiguration. To illustrate, we look once again at bind-reconfigurations, for which the sentences (4) and (5) below capture the notions of progress and effect.

$$\forall\flat\{p, o\} \cdot (\langle p, o\ \mathsf{bind}\ c\rangle \mathit{true}) \Leftrightarrow \mathsf{close\text{-}enough}(p, o) \wedge \mathsf{free}(p) \tag{4}$$

$$\forall\flat\{p, o\} \cdot [p, o\ \mathsf{bind}\ c]c = \mathsf{just}(p) \wedge \mathsf{brane}(c) = \mathsf{brane}(o) \tag{5}$$

Note that, unlike p and o, which are bound by $\forall\flat$, the variable c is bound by the modal operators $\langle p, o\ \mathsf{bind}\ c\rangle$ and $[p, o\ \mathsf{bind}\ c]$. This is useful because we expect c, on the one hand, to be interpreted in the new configuration generated as a result of binding p to o – instead of where p and o interact – and on the other hand, to be tracked through reconfigurations along with the specific protein p and organelle o that have led to its formation.

Typed modalities – and their corresponding modal operators, as in (4) and (5) – also provide a scope for addressing the property of binding proteins to organelles discussed in Section 2, which we can rewrite as follows:

$$\forall\flat\{p, o\} \cdot \mathsf{close\text{-}enough}(p, o) \wedge \mathsf{free}(p) \Rightarrow [p\, o\ \mathsf{bind}\ c]\ \mathsf{part\text{-}of}(p, c) \wedge \mathsf{brane}(c) = \mathsf{brane}(o). \tag{6}$$

This sentence is a consequence of the above axiomatization of membrane budding. More specifically, one can easily see that any network that satisfies (2) and (5) satisfies (6) too. We make this relationship precise in Section 4, where we also outline a general and fully automated technique for checking entailment.

The entire DNI-specification of membrane budding, which covers many more static constraints and requirements of the biological model as well as a full description of each of the seven types of reconfigurations, is available in [29].

4 A proof-by-translation technique for DNI

In order to check properties of dynamic networks of interactions, we employ a proof-by-translation technique whereby both the specification of the system that we intend to analyse and the property to be checked are translated to other logical systems for which tool support for formal verification has already been developed. In our case, the logical system of reference into which we translate DNI-specifications is (many-sorted) partial first-order logic (PFOL) [6].

The main feature of PFOL is that it accommodates both total and partial operations. Hence, a PFOL-signature can be broadly described as a first-order signature $\langle S, F, P\rangle$ where F is partitioned in two families TF and PF of sets of total- and partial-operation symbols, respectively. We denote such signatures by $\langle S, TF, PF, P\rangle$. Models and sentences are defined in a similar manner to ordinary first-order logic, except that (a) the operation symbols in PF are interpreted as partial rather than total functions, and (b) quantifiers apply only to total first-order variables. The notion of satisfaction of a sentence by a model needs to be adapted accordingly, but only for equational and relational atoms. We take the *existential* approach by which a PFOL-model M satisfies an equational atom $t_1 = t_2$ when the interpretations M_{t_1} and M_{t_2} are both defined and are equal; likewise, M satisfies $\pi(t)$ when M_t is defined and $M_t \in M_\pi$.

The gist of the translation of DNI into PFOL lies in making network configurations explicit and in encoding typed modalities as many-sorted relations:

- every DNI-signature $\Xi = \langle \Delta, \Omega, N, \Lambda\rangle$ is encoded as a PFOL-signature $\Phi(\Xi) = \langle S_\Xi, TF_\Xi, PF_\Xi, P_\Xi\rangle$ together with a set Γ_Ξ of $\Phi(\Xi)$-sentences;
- every Ξ-sentence ρ is translated to a PFOL-sentence $\alpha(\rho)$[5] over $\Phi(\Xi)$; and
- every $\Phi(\Xi)$-model M' that satisfies Γ_Ξ is reduced to a Ξ-network $\beta(M')$[5].

The encoding of signatures. We let $S_\Xi = S \cup \{\varkappa\}$, where S is the set of sorts of Ξ and $\varkappa$ is a new sort, distinct from those in S, that corresponds to configurations. Now suppose that $\Delta = \langle S, F_\Delta, P_\Delta\rangle$ and $\Omega = \langle S, F_\Omega, P_\Omega\rangle$. The remaining components of $\Phi(\Xi)$ are defined as follows:

$$TF_\Xi = F_\Delta \uplus \{i\colon \varepsilon \to \varkappa \mid i \in N\} \qquad \text{(encoding nominals as constants of sort } \varkappa)$$

$$PF_\Xi = \{\sigma\colon \varkappa\, u \to s \mid (\sigma\colon u \to s) \in F_\Omega\} \qquad \text{(making configs explicit for local operations)}$$

$$P_\Xi = \{\mathrm{loc}\colon \varkappa\, s \mid s \in S\} \qquad \text{(encoding the 'locality' of network components)}$$

$$\uplus\, P_\Delta \uplus \{\pi\colon \varkappa\, u \mid (\pi\colon u) \in P_\Omega\} \qquad \text{(making configs explicit for local predicates)}$$

$$\uplus\, \{\lambda\colon \varkappa\, u\, \varkappa\, v \mid (\lambda\colon u \times v) \in \Lambda\} \qquad \text{(encoding typed modalities as relations)}$$

[5] Technically, the maps α and β are both indexed by the DNI-signature under consideration, but we put this detail aside in order to reduce the notational burden.

The set Γ_Ξ of $\Phi(\Xi)$-sentences contains axioms ensuring, for example, that local operations are well defined (in 7, which applies to all symbols $\sigma\colon u \to s$ in F_Ω), that shared operations indeed agree on shared local components (in 8, which applies to symbols $\varsigma\colon u \to s$ in $F_\Delta \cap F_\Omega$), and that transitions involve triggers that are local to their source configuration and outcomes that are local to their target configuration (in 9, for all modalities $\lambda\colon u \rightarrowtail v$ in Λ). To that end, we let k and k' be variables of sort $\varkappa$, x and y be tuples of variables corresponding to arities u and v, respectively, and we write $\mathrm{loc}(k, x)$ in place of $\mathrm{loc}(k, x_1) \wedge \cdots \wedge \mathrm{loc}(k, x_n)$ when $x = (x_1, \ldots, x_n)$. Similar axioms to (7) and (8) are added for local and for shared predicates. We also use the derived PFOL-atom $\mathrm{def}\, t$, which stands for $t = t$, indicating that (the interpretation of) the term t is defined.

$$\forall \{k, x\} \cdot \big(\mathrm{def}\, \sigma(k, x) \Rightarrow \mathrm{loc}(k, x)\big) \wedge \big(\mathrm{loc}(k, x) \Rightarrow \mathrm{loc}(k, \sigma(k, x))\big) \tag{7}$$

$$\forall \{k, x\} \cdot \mathrm{loc}(k, x) \Rightarrow \varsigma(k, x) = \varsigma(x) \tag{8}$$

$$\forall \{k, k', x, y\} \cdot \lambda(k, x, k', y) \Rightarrow \mathrm{loc}(k, x) \wedge \mathrm{loc}(k', y) \tag{9}$$

The translation of sentences. We let $\alpha(\rho) = \forall \{k\} \cdot \alpha(k, \rho)$, where $\alpha(k, \rho)$ is defined inductively on the structure of the DNI-sentence ρ, as follows:

- $\alpha(k, t_1 = t_2) = \big(\alpha(k, t_1) = \alpha(k, t_2)\big)$, where $\alpha(k, \sigma(t)) = \sigma(k, \alpha(k, t))$ if σ is a local operation symbol, and $\alpha(k, \sigma(t)) = \sigma(\alpha(k, t))$ if σ is not local;[6]
- $\alpha(k, \pi(t)) = \pi(k, \alpha(k, t))$ if π is local, and $\alpha(k, \pi(t)) = \pi(\alpha(k, t))$ otherwise;
- $\alpha(k, \flat l) = \mathrm{loc}(k, \alpha(k, l))$;
- $\alpha(k, i) = (k = i)$, when i is a nominal;
- $\alpha(k, \neg\rho) = \neg\alpha(k, \rho)$ and $\alpha(k, \rho_1 \wedge \rho_2) = \alpha(k, \rho_1) \wedge \alpha(k, \rho_2)$;
- $\alpha(k, @i \cdot \rho) = \alpha(i, \rho)$;
- $\alpha(k, [t\ \lambda\ z]\rho') = \forall \{k', z\} \cdot \lambda(k, \alpha(k, t), k', z) \Rightarrow \alpha(k', \rho')$;
- $\alpha(k, \forall X; K \cdot \rho^\dagger) = \forall (X \cup K) \cdot \alpha(k, \rho^\dagger)$.

The reduction of models. Suppose M' is a PFOL-model of $\Phi(\Xi)$ that satisfies Γ_Ξ. We define $\beta(M')$ as the Ξ-network $\langle D, W, M\rangle$ given by:

- $D = M'{\restriction_\Delta}$ – for this purpose, notice that Δ is a subsignature of $\Phi(\Xi)$;
- $|W| = M'_\varkappa$, $W_i = M'_i$ for all nominals $i \in N$, and $W_\lambda = \{(w, w') \mid (w, a, w', a') \in M'_\lambda$ for some $a \in M'_u$ and $a' \in M'_v\}$ for all modalities $\lambda\colon u \rightarrowtail v$;
- for every $w \in |W|$, $M_{w,s} = \{a \in M'_s \mid (w, a) \in M'_{\mathrm{loc}\colon \varkappa s}\}$ for all sorts $s \in S$, $M_{w,\sigma}(a) = M'_\sigma(w, a)$ for all operation symbols $\sigma\colon u \to s$ in Ω and $a \in M_{w,u}$, and $M_{w,\pi} = \{a \in M_{w,u} \mid (w, a) \in M'_\pi\}$ for all predicate symbols $\pi\colon u$ in Ω;
- for every modality $\lambda\colon u \rightarrowtail v$ and every transition $(w, w') \in W_\lambda$, we have $M_{w\lambda w'} = \{(a, a') \in M_{w,u} \times M_{w',v} \mid (w, a, w', a') \in M'_\lambda\}$.

[6] When $t = (t_1, \ldots, t_n)$ is a tuple of terms, by $\alpha(k, t)$ we mean $(\alpha(k, t_1), \ldots, \alpha(k, t_n))$.

It is straightforward to check now, based on our initial assumption that M' satisfies the constraints in Γ_Ξ, that $\beta(M')$ is indeed a well defined Ξ-network.

Proposition 1. *For every Ξ-sentence ρ and $\Phi(\Xi)$-model M' that satisfies Γ_Ξ,*

$$M' \models^{PFOL}_{\Phi(\Xi)} \alpha(\rho) \quad \textit{if and only if} \quad \beta(M') \models^{DNI}_{\Xi} \rho.^{7}$$

That is, the syntactic and the semantic components of the proposed encoding of DNI into PFOL are mutually coherent. Moreover, it is easy to see that the encoding is *conservative* in the sense that for every Ξ-network $\langle D, W, M\rangle$ there exists a $\Phi(\Xi)$-model M' such that $\beta(M') = \langle D, W, M\rangle$ – by which we also mean that M' satisfies all sentences in Γ_Ξ. Used in conjunction with Proposition 1, being conservative ensures that the semantic entailment relations of DNI are indistinguishable from the semantic entailment relations of its encoding in PFOL. We write $E \models^{\text{DNI}}_{\Xi} \rho$ to indicate that ρ is a semantic consequence of E; i.e., ρ holds in all networks of interactions that globally satisfy all sentences in E.

Proposition 2. *For every set E of Ξ-sentences and every Ξ-sentence ρ,*

$$E \models^{DNI}_{\Xi} \rho \quad \textit{if and only if} \quad \alpha(E) \cup \Gamma_\Xi \models^{PFOL}_{\Phi(\Xi)} \alpha(\rho).$$

A preliminary analysis of membrane budding. The practical significance of Proposition 2 is that any sound and complete syntactic entailment system for PFOL yields a sound and complete syntactic entailment system for DNI, which enables us to conduct proofs in DNI using syntactic entailment systems and supplementary tools 'borrowed' from PFOL – cf. [5].

We use this result to analyse the specification of membrane budding developed in Examples 1 and 2, checking whether it guarantees the property (6) of binding proteins to organelles that we originally discussed in Section 2. We can even strengthen the requirement by asking for the coat c to be newly generated through binding. For this purpose, we rewrite the proof goal as follows:

$$E_{\mathsf{MB}} \models \downarrow k \cdot \forall \flat\{p, o\} \cdot \mathsf{close\text{-}enough}(p, o) \wedge \mathsf{free}(p) \\ \Rightarrow [p\, o\, \mathsf{bind}\ c]\, \mathsf{part\text{-}of}(p, c) \wedge @k \cdot \neg \flat c \wedge \mathsf{brane}(c) = \mathsf{brane}(o).$$

The actual proof is performed, according to Proposition 2, in two stages: first we translate the proof goal to PFOL, then we use HETS [23] along with automated first-order theorem provers such as E [28], SPASS [31], and Vampire [26] to further analyse it – the way we combine provers is similar to the Sledgehammer tool [3] for Isabelle/HOL. The same technology is employed to check a much more complex requirement, namely that cargo molecules can indeed be transported between organelles through the stages of membrane budding. Using typed modalities, this property can be formalized as a DNI-sentence of the form:

[7] Here, we annotate the double-turnstile symbols with DNI and PFOL in order to distinguish between the two types of satisfaction relation used in the statement.

$$\mathsf{enclosed}(x, \mathsf{brane}(os)) \Rightarrow \langle p_j\ os\ \mathsf{bind}\ c_j\rangle \ldots \langle p_1\ x\ \mathsf{capture}\ \rangle \ldots \langle c_1\ c_2\ \mathsf{merge}\ c\rangle \ldots$$
$$\langle c\ \mathsf{bud}\ b\rangle \ldots [b\ \mathsf{split}\ v] \ldots \langle p_j\ v\ \mathsf{uncoat}\ \rangle \ldots \langle v\ ot\ \mathsf{fuse}\ \rangle \mathsf{enclosed}(x, \mathsf{brane}(ot))$$

where x is a variable of sort Cargo, os and ot are variables of sort Organelle (corresponding to the source and target organelle, respectively), p_j are variables of sort Protein, c and c_j are variables of sort Coat, b is a Bud and, lastly, v is a Vesicle. The full formalization of this requirement is available in [29].

5 Conclusions

In this paper, we have presented an extension of the conventional modal-logic approach to reconfigurable systems, giving prominence to the interactions that trigger reconfigurations (through typed modalities) and to the structural changes that they induce (through varying quantification domains and flexible attributes). In addition, we have shown how the standard translation of modal into first-order logic can be adapted for dynamic networks of interactions, providing in this way a sound and complete proof-by-translation technique for specifications and properties written using the formalism advanced in Section 3. To illustrate the contributions of our work, we have formalized the biological process of membrane budding – a non-trivial example of reconfigurable system that raises significant challenges to previously developed modal specification languages.

The full analysis of membrane budding, which we have only touched upon in this paper, has revealed the need for further enhancements of our specification and verification technology. The proof goals we considered in Section 4 are pairs consisting of a monolithic theory presentation (say, of membrane budding) and a DNI-property whose entailment is scrutinized. Because we rely on automated theorem provers, this approach works well for simple verification tasks, but it quickly becomes impractical in large-scale case studies. To carry out the analysis of membrane budding in [29], we made use of standard modularization techniques from institution theory [27] both at the specification and at the verification stage; the original analysis task is thus reduced to smaller proof goals, which are easier to discharge automatically. We aim to further investigate the institution-theoretic foundations of this method in the context of DNI.

In the long run, through DNI we seek to develop a formal specification framework based on modal logic and typed modalities for reasoning about systems that exhibit interactive behaviour and whose configurations change dynamically as a result of interactions. Besides modal logic and its hybrid variants, there are numerous other approaches to reconfigurable systems based on graph or term rewriting, action calculi, tile logic [15] and, notably, bigraphs [22] – which inspired us to study membrane budding in the first place. Therefore, another natural next step is to study the relationship between DNI and logics such as BiLog [9], which are effective at capturing bigraphs and their substructures.

References

1. Blackburn, P.: Representation, reasoning, and relational structures: A hybrid logic manifesto. Logic Journal of IGPL **8**(3), 339–365 (2000)
2. Blackburn, P., Marx, M.: Tableaux for quantified hybrid logic. In: TABLEAUX 2002. Lecture Notes in Computer Science, vol. 2381, pp. 38–52. Springer (2002)
3. Blanchette, J.C., Böhme, S., Paulson, L.C.: Extending sledgehammer with SMT solvers. Journal of Automated Reasoning **51**(1), 109–128 (2013)
4. Braüner, T.: First-order hybrid logic: introduction and survey. Logic Journal of IGPL **22**(1), 155–165 (2014)
5. Cerioli, M., Meseguer, J.: May I borrow your logic? (transporting logical structures along maps). Theoretical Computer Science **173**(2), 311–347 (1997)
6. Cerioli, M., Mossakowski, T., Reichel, H.: From total equational to partial first-order logic. In: Algebraic Foundations of Systems Specification, pp. 31–104. IFIP State-of-the-Art Reports, Springer (1999)
7. Clarke, D.: A basic logic for reasoning about connector reconfiguration. Fundamenta Informaticae **82**(4), 361–390 (2008)
8. Codescu, M.: Hybridisation of institutions in Hets. In: CALCO 2019. LIPIcs, vol. 139, pp. 17:1–17:10. Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2019)
9. Conforti, G., Macedonio, D., Sassone, V.: Static BiLog: a unifying language for spatial structures. Fundamenta Informaticae **80**(1-3), 91–110 (2007)
10. Diaconescu, R.: Introducing H, an institution-based formal specification and verification language. Logica Universalis **14**(2), 259–277 (2020)
11. Diaconescu, R., Madeira, A.: Encoding hybridized institutions into first-order logic. Mathematical Structures in Computer Science **26**(5), 745–788 (2016)
12. Fiadeiro, J.L., Lopes, A.: A model for dynamic reconfiguration in service-oriented architectures. Software and Systems Modeling **12**(2), 349–367 (2013)
13. Fiadeiro, J.L., Ţuţu, I., Lopes, A., Pavlovic, D.: Logics for actor networks: A two-stage constrained-hybridisation approach. Journal of Logical and Algebraic Methods in Programming **106**, 141–166 (2019)
14. Fitting, M., Mendelsohn, R.L.: First-Order Modal Logic. Kluwer Academic Publishers (1998)
15. Gadducci, F., Montanari, U.: Comparing logics for rewriting: rewriting logic, action calculi and tile logic. Theoretical Computer Science **285**(2), 319–358 (2002)
16. Garson, J.W.: Quantification in modal logic. In: Handbook of Philosophical Logic. Springer Netherlands (2001)
17. Goguen, J.A., Burstall, R.M.: Institutions: abstract model theory for specification and programming. Journal of the ACM **39**(1), 95–146 (1992)
18. Hennicker, R., Madeira, A., Knapp, A.: A hybrid dynamic logic for event/data-based systems. In: FASE 2019. Lecture Notes in Computer Science, vol. 11424, pp. 79–97. Springer (2019)
19. Kirchhausen, T.: Three ways to make a vesicle. Nature reviews. Molecular cell biology pp. 187–198 (2000)
20. Krivine, J., Milner, R., Troina, A.: Stochastic bigraphs. In: MFPS 2008. Electronic Notes in Theoretical Computer Science, vol. 218, pp. 73–96. Elsevier (2008)
21. Madeira, A., Neves, R., Barbosa, L.S., Martins, M.A.: A method for rigorous design of reconfigurable systems. Science of Computer Programming **132**, 50–76 (2016)

22. Milner, R.: The Space and Motion of Communicating Agents. Cambridge University Press (2009)
23. Mossakowski, T., Maeder, C., Lüttich, K.: The heterogeneous tool set (Hets). In: The 4th International Verification Workshop. CEUR Workshop Proceedings, vol. 259. CEUR-WS.org (2007)
24. Pratt, V.R.: Semantical considerations on Floyd-Hoare logic. In: 17th Annual Symposium on Foundations of Computer Science. pp. 109–121. IEEE Computer Society (1976)
25. Regot, S., Macía, J., Conde-Pueyo, N., Furukawa, K., Kjellén, J., Peeters, T., Hohmann, S., de Nadal, E., Posas, F., Solé, R.V.: Distributed biological computation with multicellular engineered networks. Nature **469**(7329), 207–211 (2011)
26. Riazanov, A., Voronkov, A.: The design and implementation of Vampire. AI Communications **15**(2-3), 91–110 (2002)
27. Sannella, D., Tarlecki, A.: Foundations of Algebraic Specification and Formal Software Development. Monographs in Theoretical Computer Science. An EATCS Series, Springer (2011)
28. Schulz, S., Cruanes, S., Vukmirovic, P.: Faster, higher, stronger: E 2.3. In: Fontaine, P. (ed.) Automated Deduction – CADE 27. Lecture Notes in Computer Science, vol. 11716, pp. 495–507. Springer (2019)
29. Ţuţu, I., Chiriţă, C.E., Fiadeiro, J.L.: A DNI-specification of membrane budding. Ontohub (2021), https://ontohub.org/dni
30. Ţuţu, I., Chiriţă, C.E., Lopes, A., Fiadeiro, J.L.: Logical support for bike-sharing system design. In: From Software Engineering to Formal Methods and Tools, and Back. Lecture Notes in Computer Science, vol. 11865, pp. 152–171. Springer (2019)
31. Weidenbach, C., Dimova, D., Fietzke, A., Kumar, R., Suda, M., Wischnewski, P.: SPASS version 3.5. In: Schmidt, R.A. (ed.) Automated Deduction – CADE 22. Lecture Notes in Computer Science, vol. 5663, pp. 140–145. Springer (2009)