Layered logics, coalgebraically

Luís Soares Barbosa

HASLab INESC TEC & UNU-EGOV, United Nations University
Campus de Couros, Guimarães, Portugal
lsb@di.uminho.pt

Abstract. This short note revisits layered logics from a coalgebraic point of view, and proposes a naturality condition to express the typical hierarchical requirement under which all abstract transitions should be traceable in more specialised layers.

Keywords: layered logics; hierarchical models; coalgebra.

1 Introduction

A plethora of logics is used in Software Engineering to support the specification of systems' requirements and properties, as well as to verify whether, or to what extent, they are enforced in specific implementations. Broadly speaking, the logics of dynamical systems are *modal*, *i.e.* they provide operators which qualify formulas as holding in a certain *mode*. In mediaeval Scholastics such modes represented the strength of assertion (e.g. 'necessity' or 'possibility'). In temporal reasoning they can refer to a future or past instant, or a collection thereof. Similarly, one may express epistemic states (e.g. 'as everyone knows'), deontic obligations (e.g. 'when legally entitled'), or spatial states (e.g. 'in every point of a surface'). Regarding dynamical systems as transformations of state spaces according to specific transition shapes, *i.e.* as coalgebras for particular functors [7] such modes refer to particular configurations of successor states as defined, or induced, by the coalgebra dynamics. Coalgebra provides a *uniform* characterisation inducing 'canonical' notions of modality and the corresponding logic with respect to the underlying functor[3]. General questions in modal logic, such as the trade-off between expressiveness and computational tractability, or the relationship between logical equivalence and bisimilarity, can be addressed at this (appropriate) level of abstraction.

In this sense, modal logic is essentially coalgebraic. Its classical extensions, for example hybrid logic, which is able to pinpoint specific states and index to them the satisfaction relation, can also be easily accommodated in the framework.

This short note revisits a logic suitable to express properties of, and reason about, $\mathfrak{n}$-layered, hierarchical transition systems, from a coalgebraic perspective, building on the team previous results reported in references [5, 6]. In particular it is shown how the *hierachical condition*, informaly stated under the *motto* 'upper transitions should be traceable in the layer below' can be expressed as a naturality condition in the models.

2 Reasoning about hierarchical designs

Hierarchical transition systems are a popular mathematical structure to represent state-based software applications in which different layers of abstraction are captured by interrelated state machines. The decomposition of high-level states into inner sub-states, and of their transitions into inner sub-transitions, is a common refinement procedure adopted in a number of specification formalisms.

In a recent paper [5] the author and his collaborators proposed an hybrid layered logic to reason about (non deterministic) transition systems. The diagram in Fig. 1, representing a partial view of a strongbox controller, is taken from that paper as an illustration of the sort of examples we have in mind.

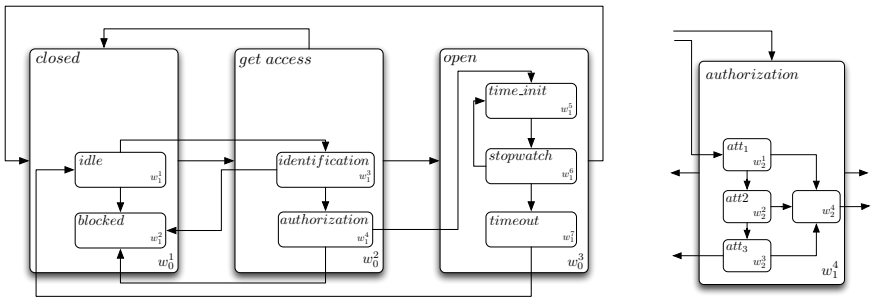


Fig. 1. An hierarchical transition system.

The strongbox controller is specified at three different levels of abstraction, expressing the progressive decomposition not only of its internal states, but also of its transitions. Thus, each 'high-level' state gives rise to a new, local transition system, and each 'upper-level' transition is decomposed into a number of 'intrusive' transitions from sub-states of the 'lower-level' transition system corresponding to the refinement of the original source state, to sub-states of the corresponding refinements of original target states. For instance, the (upper) close state can be refined into a (inner) transition system with two (sub) states: one, idle, representing the system waiting for the order to proceed for the get access state, and another one, blocked, capturing a system which is unable to proceed with the opening process (e.g. when authorised access for a given user was definitively denied). In this scenario, the upper level transition from closed to get access can be realised by, at least, one intrusive transition between the closed sub-state idle and the get access sub-state identification, in which the user identification is to be checked before proceeding. This refinement is illustrated in the the left part of Fig. 1.

The logic proposed in [5] to reason about this sort of systems is modal (so that state transitions can be expressed), combined with hybrid features to refer to specific, individual states. The qualifier *hybrid* [1, 2] refers to an extension of modal languages with symbols, called *nominals*, which explicitly refer to individual states in the underlying Kripke structure. A satisfaction operator $@_i\varphi$ stands for φ holding in the state named by nominal i.

Signatures are n-families of disjoint, possible empty, sets of symbols

$$\Delta^n = \big(\mathrm{Prop}_k, \mathrm{Nom}_k\big)_{k\in\{0,\cdots,n\}}\,.$$

For example, to specify the strongbox above, one considers a signature Δ^2 for the three layers presented, numbered from 0 (the most abstract) to 2. The set of formulas $\mathrm{Fm}(\Delta^n)$ is the n-family recursively defined, for each k, by

$$\varphi_0 \ni i_0 \mid p_0 \mid \neg\varphi_0 \mid \varphi_0 \wedge \varphi_0 \mid @_{i_0}\varphi_0 \mid \Box_0\varphi_0$$
$$\varphi_0^b \ni i_0 \mid p_0 \mid @_{i_0}\varphi_0 \mid \Box_0\varphi_0$$

and

$$\varphi_k \ni \varphi_{k-1}^b \mid i_k \mid p_k \mid \neg\varphi_k \mid \varphi_k \wedge \varphi_k \mid @_{i_k}\varphi_k \mid \Box_k\varphi_k$$

where for any $k \in \{1,\ldots,n\}$, the basic formulas are defined by

$$\varphi_{k-1}^b \ni i_{k-1} \mid p_{k-1} \mid \varphi_{k-2}^b \mid @_{i_{k-1}}\varphi_{k-1} \mid \Box_{k-1}\varphi_{k-1}$$

for $k \in \{2,\cdots,n\}$, $p_k \in \mathrm{Prop}_k$ and $i_k \in \mathrm{Nom}_k$.

This language is able to express properties of very different natures. For instance, one may express inner-outer relations between named states (e.g. $@_{\mathtt{idle}_1}\mathtt{closed}_0$ or $@_{\mathtt{att}_{1\,2}}\mathtt{open}_0$) as well as a variety of transitions. Those include, for example, the layered transition $@_{\mathtt{get\_access}_0}\diamond_0\,\mathtt{open}_0$, the 0-internal transition $@_{\mathtt{identification}_1}\diamond_1\,\mathtt{authorisation}_1$ or intrusive transitions like $@_{\mathtt{idle}_1}\diamond_1\,\mathtt{authorisation}_1$ and $\mathtt{get\_access}_0 \to \diamond_1\mathtt{open}_0$.

3 ... coalgebraically

The whole programme can actually be carried out in a coalgebraic setting. The basic observation is that when defining a model for this logic the family of accessibility relations considered in [5] is replaced by a family of coalgebras for the same endofunctor, each of which captures the dynamics of the appropriate layer.

Thus, a n-layered model $M \in \mathrm{Mod}^n(\Delta^n)$ is a tuple

$$M = \langle W^n, D^n, \alpha^n, V^n \rangle$$

where $W^n = (W_k)_{k\in\{0,\cdots,n\}}$ is a family of disjoint sets of states, and $D^n \subseteq W_0 \times \cdots \times W_n$ is a definition predicate that singles out the chains of states across the n levels which are considered meaningful 'global' states. Denoting by D_k the k-restriction $D^n|_k$ to the first $k+1$ columns, for each $k \in \{0, \cdots, n\}$, it is the case that

$$W_k = \{v_k | D_k\langle w_0, \cdots, w_{k-1}, v_k\rangle, \text{ for some } w_0, \cdots, w_{k-1} \text{ such that } D_{k-1}\langle w_0, \cdots, w_{k-1}\rangle\}\,.$$

The 'dynamics': $\alpha^n = \big(\alpha_k : D_k \longrightarrow \mathcal{F}(D_k)\big)_{k\in\{0,\cdots,n\}}$ is a family of $\mathcal{F}$-coalgebras specifying the system's evolution at each level in the hierarchy. Finally, $V^n = (V_k^{\mathrm{Prop}}, V_k^{\mathrm{Nom}})_{k\in\{0,\cdots,n\}}$ is a family of pairs of valuations defined as one could expect: $V_k^{\mathrm{Prop}} : \mathrm{Prop}_k \to \mathcal{P}(D_k)$, and $V_k^{\mathrm{Nom}} : \mathrm{Nom}_k \to W_k$.

The advantage of expressing the transition structure coalgebraically is the genericity of the approach. Actually, making $\mathcal{F} = \mathcal{P}$, the powerset monad, we are brought back the usual Kripke structure, modal formulas being interpreted over a non deterministic transition system. Different alternatives can be considered by varying $\mathcal{F}$. For example, modalities can be interpreted in a probabilistic setting by instantiating $\mathcal{F}$ with the sub-distribution monad $\mathcal{D}_{\leq}(X) = \{\mu : X \to \mathbb{R}_{\geq 0} |\ \sum_{x\in X} \mu x \leq 1\}$ which captures probabilistic transitions — note that what is missing to 1 above can be seen as the probability of some sort of 'systemic' failure, such as deadlock, to occur.

As expected, the satisfaction relation is a family $\models^n = (\models_k)_{k\in\{0,\cdots,n\}}$ defined, for each $w_r \in W^r$, $r \in \{0, \cdots, k\}$, $k \leq n$, such that $D_k\langle w_0, \cdots w_k\rangle$. The case of interest in the context of this note is the one for modalities, *i.e.* $M_k, w_0, \cdots, w_k \models_k \Box_k \varphi_k$ iff

$$\forall_{v_0\in W_0,\cdots,v_k\in W_k}.\ \langle v_0, \cdots, v_k\rangle \in \alpha_k\langle w_0, \cdots, w_k\rangle \text{ implies } M, v_0, \cdots, v_k \models_k \varphi_k\,.$$

The hybrid part is given by

- $M_k, w_0, \cdots, w_k \models_k i_k$ iff $w_k = V_k^{\mathrm{Nom}}(i_k)$ and $D_k\langle w_0, \cdots, w_{k-1}, V_k^{\mathrm{Nom}}(i_k)\rangle$,
- $M_k, w_0, \cdots, w_k \models_k @_{i_k}\varphi_k$ iff $M_k, w_0, \cdots w_{k-1}, V_k^{\mathrm{Nom}}(i_k) \models_k \varphi_k$ and $D_k\langle w_0, \cdots w_{k-1}, V_k^{\mathrm{Nom}}(i_k)\rangle$.

The Boolean part, finally, is defined as usual, just taking care of the definability interdependence captured by D^n. The only aspect one needs to take into account is the interplay between the satisfaction operators and the modalities induced (or built over) the coalgebra. For example, one has to specify that a formula like $@_i\ \varphi$ must be valid either in the whole model or nowhere. In an Hilbert calculus this can be achieved through an extra axiom, for each modal operator $\divideontimes$:

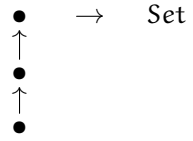
$$@_i\varphi \ \Rightarrow\ (\divideontimes(\varphi_1, \cdots, \varphi_k) \ \Leftrightarrow\ \divideontimes(\varphi_1 \wedge @_i\varphi, \cdots, \varphi_k \wedge @_i\varphi)$$

capturing the intended validity of $@_i\varphi$ irrespective to the interpretation of each φ_j.

As mentioned in the Introduction, there is a specific, particularly well-behaved class of layered models, called *hierarchical*, in which all upper transitions are traceable in the layer below. Technically, this amounts to the requirement that the restriction of a coalgebra α_k to the state space of α_{k-1} coincides with the latter. In this case, the family of coalgebras α^n is called *hierarchically compatible*.

The example sketched in Fig. 1, is clearly an hierarchical model. Examples of non-hierarchical layered models can be achieved by removing some 0-transitions depicted in the diagram above (e.g. the one linking the named states $\mathtt{closed}_0$ and $\mathtt{get\_access}_0$). The hierarchical condition is quite natural and somehow inherent to well-known design formalisms such as D. Harel's statecharts [4] and the subsequent UML hierarchical state machines, among others.

What is worth to notice is that the *hierarchical* requirement can be expressed as a naturality condition as follows. The first step is to regard the family of coalgebras α^n as a coalgebra in a *functor category*, for a suitable finite chain:

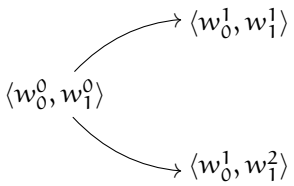


Arrows (thus, component coalgebras) are families of natural transformations, making the following diagram to commute for all k,

$$\begin{array}{ccc} D_k & \xrightarrow{\alpha_k} & \mathcal{F}(D_k) \\ {\scriptstyle \pi_k}\downarrow & & \downarrow{\scriptstyle \mathcal{F}(\pi_k)} \\ D_{k-1} & \xrightarrow{\alpha_{k-1}} & \mathcal{F}(D_{k-1}) \end{array}$$

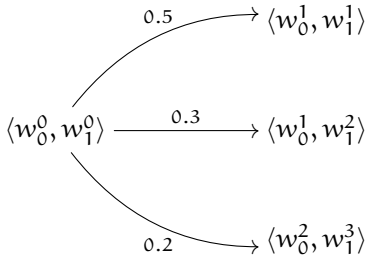
where $\pi_k : D_k \longrightarrow D_{k-1}$ be given by $\pi_k \langle w_0, \cdots, w_{k-1}, w_k \rangle \mathrel{\widehat{=}} \langle w_0, \cdots, w_{k-1} \rangle$.

Let us illustrate this construction. For $\mathcal{F} = \mathcal{P}$, transitions

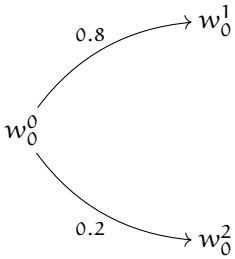


exist at level 1 iff a transition $w_0^0 \longrightarrow w_0^1$ exists at level 0.

For another example, consider $\mathcal{F} = \mathcal{D}_{\leq}$. In an hierarchical (probabilistic) system naturality entails that the existence of transitions



requires



at level 0.

4 Concluding

As it happens in other domains of Computer Science, also at this level of 'logics engineering' Coalgebra is a source of genericity. Clearly, whenever functor $\mathcal{F}$ is a monad, the monadic structure can be taken as the basis for a model algebra in which the composition of hierarchical systems can be addressed. Canonical formats for bisimulation come for free, and often Hennessy-Milner like theorems can be proved in a generic setting.

On the other hand, the *motto* for hierarchical models – 'upper transitions should be traceable in the layer below' – can be re-phrased, in a more precise way, as *layer compatibility is natural*. This, again, is a source of genericity: the very notion of *hierarchical* is made relative to whatever poset P is used to define the *layer structure* intended to be respected. We leave the reader of this short note with the (easy) quiz of re-phrasing this notion for the four different posets depicted in Fig. 2.

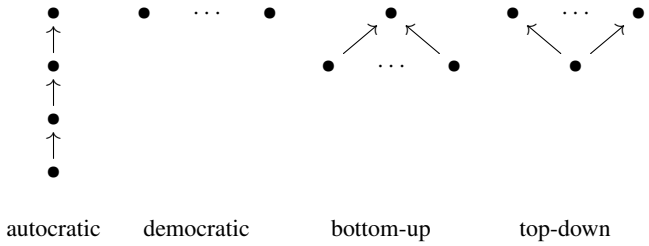


Fig. 2. Some shapes for layered structures.

Aknowlegdments. This paper is a result of the project *SmartEGOV: Harnessing EGOV for Smart Governance*, NORTE-01-0145-FEDER-000037, supported by Norte Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, through the European Regional Development Fund (EFDR).

References

1. P. Blackburn. Representation, reasoning, and relational structures: a hybrid logic manifesto. *Logic Journal of IGPL*, 8(3):339–365, 2000.
2. T. Brauner. *Hybrid Logic and its Proof-Theory*. Applied Logic Series. Springer, 2010.
3. C. Cîrstea, A. Kurz, D. Pattinson, L. Schröder, and Y. Venema. Modal logics are coalgebraic. *Comput. J.*, 54(1):31–41, 2011.
4. D. Harel. Statecharts: A visual formalism for complex systems. *Sci. Comput. Program.*, 8(3):231–274, 1987.
5. A. Madeira, M. A. Martins, and L. S. Barbosa. A logic for n-dimensional hierarchical refinement. In John Derrick, Eerke A. Boiten, and Steve Reeves, editors, *Proceedings 17th International Workshop on Refinement, Refine@FM 2015, Oslo, Norway, 22nd June 2015*, volume 209 of *EPTCS*, pages 40–56, 2016.
6. A. Madeira, M. A. Martins, L. S. Barbosa, and R. Hennicker. Refinement in hybridised institutions. *Formal Aspects of Computing*, pages 1–21, 2014.
7. J. J. M. M. Rutten. Universal coalgebra: A theory of systems. *Theor. Comput. Sci.*, 249(1):3–80, 2000. (Revised version of CWI Techn. Rep. CS-R9652, 1996).