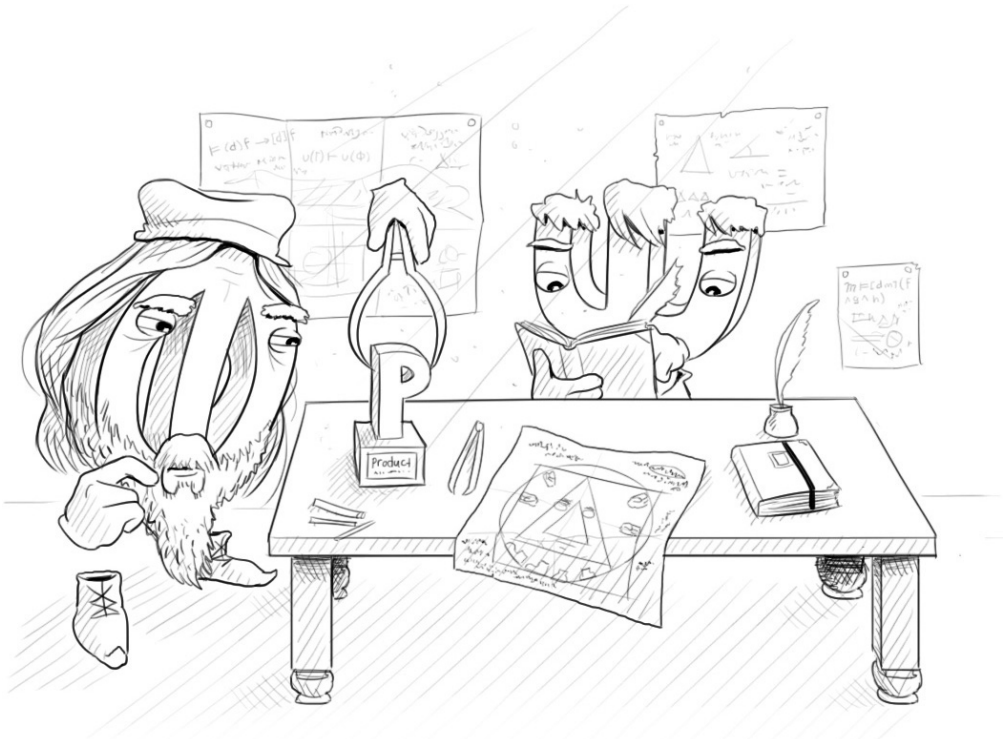
6

Delta Logic

Reasoning About Products and Delta Effects



6.1 Introduction

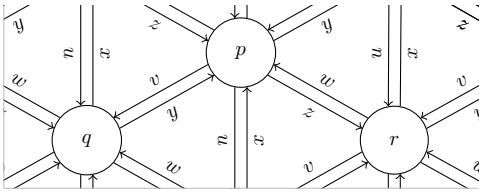


Figure 6.1: Example view of a delta frame with products p, q, r and deltas u, v, w, x, y, z currently visible.

At its core, ADM is about deltas that can transform one product into another product. But the algebraic notation of the previous chapters is not ideal for specifying and reasoning about the semantics of deltas, and what effect they have on the properties of a product. We want to be able to specify that a delta implements a specific new feature or that a delta refrains from breaking some existing feature, without talking about products. Similarly, we want to prove that certain local constraints on deltas ensure desirable global properties. This chapter introduces a modal logic tailored to this goal. For a brief introduction to modal logic, see Section 1.7.10 (page 25).

> ***Goal:*** *Create a modal logic for reasoning syntactically about the semantics of deltas and their effects on product properties.*

Basically, we take the set of products as the set of *worlds* in a frame (Figure 6.1). We then model deltas as binary relations on this set by applying semantic evaluation. The result is something very similar to dynamic logic [69]. In this logic, we want to be able to make judgments such as

$$\Vdash \langle\, d \,\rangle\, k \qquad\qquad \Vdash [\, d \,]\, k,$$

meaning "delta d *may* implement property k" (left) and "delta d *definitely* implements property k" (right). Or perhaps, if stated for all formulas ψ,

$$\Vdash \langle\, d \,\rangle\, \psi \;\to\; [\, d \,]\, \psi \qquad\qquad \Vdash [\, d \,]\, \psi \;\to\; \langle\, d \,\rangle\, \psi,$$

meaning "delta d is deterministic" (left) and "delta d is fully defined" (right). These formulas implicitly quantify over all products that d may be applied to, but such judgments may also be made with regard to specific products or models. We will also use delta models as modalities, in order to make judgments such as

$$\Vdash [dm](f \wedge g \wedge h),$$

meaning that, if it applies, delta model dm implements features f, g and h in all possible products. To that purpose, we allow the possibility of nested delta models (Section 3.6).

Section 6.2 specifies the modal language. Section 6.3 explores the modal logic on a Kripke frame level, specifying the proof theory, proving its completeness and extending it to a proof system for delta correctness. Section 6.4 explores the logic on a Kripke model level, addressing a problem in proving judgments about specific propositions. Finally, Sections 6.5 and 6.6 offer concluding remarks and discuss related work.

6.2 A Multimodal Language

One of the primary goals of this chapter is to reason about abstract delta modeling using the language and techniques of modal logic. A necessary starting point, before moving on to an axiomatic characterization (in which we are concerned with issues such as completeness), is to describe a modal language.

The first pair of example formulas on page 135 reference a property k. It comes from a set of propositional variables:

▶ **6.1. Notation (Propositional Variables):** We denote *propositional variables* by the symbols k, l, m. Sets of propositional variables are denoted by $PROP$. ⌟

We then define the language that will form the basis of our logic. The intention is to describe properties of (sets of) products in a syntactic manner, by the propositions that hold there, or those that hold in products reachable through the application of certain deltas. The language is a multimodal language (Definition 1.36) based on the specific artefacts of ADM:

▶ **6.2. Definition (Product Formulas):** Given a set of deltas (or delta models) $\mathcal{D}_{\vartriangle}$ (Section 3.6) and a set of propositional variables $PROP$, we define a multimodal language of *product formulas* with the following grammar:

$$\Psi \quad \ni \quad \varphi \quad ::= \quad \top \mid k \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle\, d \,\rangle\, \varphi$$

where $k \in PROP$ is a propositional variable and $d \in \mathcal{D}_{\vartriangle}$ is an expression resolving to a delta (model) (Sections 2.6 and 3.6). We introduce the following formulas as abbreviations, so we need only be concerned with the minimal grammar above in further analysis. For all formulas $\varphi, \psi \in \Psi$:

$$\begin{aligned}
\bot &\stackrel{\text{def}}{=} \neg\top \\
[\, d \,]\, \varphi &\stackrel{\text{def}}{=} \neg\langle\, d \,\rangle\, \neg\varphi \\
\varphi \wedge \psi &\stackrel{\text{def}}{=} \neg(\neg\varphi \vee \neg\psi) \\
\varphi \to \psi &\stackrel{\text{def}}{=} \neg\varphi \vee \psi \\
\varphi \leftrightarrow \psi &\stackrel{\text{def}}{=} (\varphi \to \psi) \wedge (\psi \to \varphi)
\end{aligned}$$

To resolve ambiguity we assume the traditional set of precedence rules (e.g. $\wedge$ binds stronger than $\vee$) and allow parentheses to override those rules.

If the set of deltas or propositional variables is not clear from context, we attach a subscript as in $\Psi_{\mathcal{D}, PROP}$. ⌟

6.3 Kripke Frames

This section defines the 'delta version' of Kripke frames and models (Definitions 1.37 and 1.39), and discusses proof theory on the frame level.

6.3.1 Kripke Semantics

Defining the Kripke semantics for a given deltoid is not difficult, because a deltoid (Definition 2.11) is already a Kripke frame (Definition 1.37). To work with delta models, a delta model closed deltoid (Definition 3.31) is assumed:

▶ **6.3. Notation (Delta Kripke Frame):** A *delta Kripke frame* $\mathfrak{F}$ is a deltoid $Dt = (\mathcal{P}, \mathcal{D}_{\triangle}, [\![\text{-}]\!])$, where the set of products $\mathcal{P}$ is the set of worlds, the set of deltas $\mathcal{D}_{\triangle}$ is the set of modal labels and the semantic evaluation operator $[\![\text{-}]\!]\colon \mathcal{D}_{\triangle} \to \mathrm{Pow}(\mathcal{P} \times \mathcal{P})$ maps each delta to a corresponding accessability relation.

For the sake of brevity, we will just write Dt or $(\mathcal{P}, \mathcal{D}_{\triangle}, [\![\text{-}]\!])$ when a delta Kripke frame is expected.

The class of all *disjunctive* delta Kripke frames is denoted $\Delta\cup\mathsf{F}$. The class of all *conjunctive* delta Kripke frames is denoted $\Delta\cap\mathsf{F}$. Both are classes of frames with an underlying set of delta (model) expressions as modalities, following their respective policies for delta model semantics (Section 3.5). ⌟

Figure 6.1 shows part of an infinite deltoid Kripke frame.

To reason about product properties, we need a *valuation function* (Definition 1.38), mapping proposition letters to the set of worlds in which they are true. A delta Kripke model is a delta Kripke frame with a valuation function:

▶ **6.4. Notation (Delta Kripke Model):** A *delta Kripke model* is a tuple $\mathfrak{M} = (Dt, \mathrm{V}) = (\mathcal{P}, \mathcal{D}_{\triangle}, [\![\text{-}]\!], \mathrm{V})$ — a deltoid Kripke frame equipped with a valuation function $\mathrm{V}\colon \mathit{PROP} \to \mathrm{Pow}(W)$. ⌟

The semantics of product formulas (Definition 6.2) can, of course, be given in the traditional manner for modal formulas: by defining a forcing relation $\Vdash$ (Definition 1.40). But in the trend set by the previous chapters, we do it instead by extending the semantic evaluation operator $[\![\text{-}]\!]$ so it can map product formulas (a syntactic notion) to sets of products (a semantic notion), which is quite compact and intuitive:

▶ **6.5. Definition (Formula Semantics):** Given a Kripke model $\mathfrak{M} = (\mathcal{P}, \mathcal{D}_{\triangle}, [\![\text{-}]\!], \mathrm{V})$, we extend semantic evaluation to product formulas as follows. For all propositional variables $k \in \mathit{PROP}$, formulas $\varphi, \psi \in \Psi$ and deltas $d \in \mathcal{D}_{\triangle}$ we define $[\![\text{-}]\!]\colon \Psi \to \mathrm{Pow}(\mathcal{P})$ by induction on the shape of the formula:

$$\begin{array}{rcl} [\![\,\top\,]\!] & \stackrel{\text{def}}{=} & \mathcal{P} \\ [\![\,k\,]\!] & \stackrel{\text{def}}{=} & \mathrm{V}(k) \\ [\![\,\varphi \vee \psi\,]\!] & \stackrel{\text{def}}{=} & [\![\,\varphi\,]\!] \cup [\![\,\psi\,]\!] \\ [\![\,\neg\varphi\,]\!] & \stackrel{\text{def}}{=} & [\![\,\top\,]\!] \setminus [\![\,\varphi\,]\!] \\ [\![\,\langle d\rangle\varphi\,]\!] & \stackrel{\text{def}}{=} & [\![\,d\,]\!]^{-1}\big([\![\,\varphi\,]\!]\big) \end{array}$$

As always, the proper subscripts can be added to disambiguate between disjunctive and conjunctive semantics. ⌟

This essentially gives us a way to describe sets of products by which properties they satisfy, including properties of which products could result if certain deltas are applied. We could reintroduce the traditional forcing relation $\Vdash$ (Definition 1.40, page 26) as follows:

6.6. Lemma: Given a deltoid Kripke model $\mathfrak{M} = (\mathcal{P}, \mathcal{D}_{\triangle}, [\![\text{-}]\!], \mathrm{V})$, product $p \in \mathcal{P}$ and formula $\varphi \in \Psi$, we have:

$$\mathfrak{M}, p \Vdash \varphi \quad \Longleftrightarrow \quad p \in [\![\,\varphi\,]\!]$$

In other words, Definition 6.5 corresponds to traditional modal semantics. □

6.3.2 Proof Theory

Now that we have a multimodal language with Kripke semantics, we define the logic that can be used to reason purely in that language, and prove that this logic is sound and complete. This then allows us to reason about the effects of deltas without resorting to semantic evaluation.

▶ **6.7. Definition (Delta Logic):** The *delta logics* $\mathbf{K\Delta\cup}$ and $\mathbf{K\Delta\cap}$ are normal modal logics[1] (Definition 1.43) generated by the following axiom schemas, which encode the laws of our algebraic operators.[2] For all delta (model) expressions $x, y \in \mathcal{D}_{\vartriangle}$ and all formulas $\varphi \in \Psi$:

- *the composition axiom*: $\langle y \cdot x \rangle \varphi \leftrightarrow \langle x \rangle \langle y \rangle \varphi \quad \in \quad \mathbf{K\Delta\cup}, \mathbf{K\Delta\cap}$
- *the choice axiom*: $\langle x \sqcup y \rangle \varphi \leftrightarrow (\langle x \rangle \varphi \vee \langle y \rangle \varphi) \quad \in \quad \mathbf{K\Delta\cup}, \mathbf{K\Delta\cap}$
- *the consensus axiom*: $\langle x \sqcap y \rangle \varphi \leftrightarrow (\langle x \rangle \varphi \wedge \langle y \rangle \varphi) \quad \in \quad \mathbf{K\Delta\cup}, \mathbf{K\Delta\cap}$
- *the neutral delta axiom*: $\varphi \leftrightarrow \langle \varepsilon \rangle \varphi \quad \in \quad \mathbf{K\Delta\cup}, \mathbf{K\Delta\cap}$
- *the empty delta axiom*: $[\bot] \varphi \quad \in \quad \mathbf{K\Delta\cup}, \mathbf{K\Delta\cap}$

Then, depending on policy regarding delta model semantics, one of the following axiom schemas should be added:

- *the delta model axiom* $\Delta\cup$: $\langle dm \rangle \varphi \leftrightarrow \bigvee_{d \in \mathrm{derv}(dm)} \langle\, d \,\rangle \varphi \quad \in \quad \mathbf{K\Delta\cup}$
- *the delta model axiom* $\Delta\cap$: $\langle dm \rangle \varphi \leftrightarrow \bigwedge_{d \in \mathrm{derv}(dm)} \langle\, d \,\rangle \varphi \quad \in \quad \mathbf{K\Delta\cap}$ ⌟

For disjunctive semantics, a different formulation for the disjunctive delta model axiom follows straightforwardly from Definition 3.25:

▶ **6.8. Theorem:** For nonempty delta model $dm = (D, \prec)$ and all formulas φ:

$$\Vdash_{\Delta\cup\mathsf{F}} \quad \langle (\varnothing, \varnothing) \rangle \varphi \leftrightarrow \varphi$$
$$\Vdash_{\Delta\cup\mathsf{F}} \quad \langle dm \rangle \varphi \leftrightarrow \bigvee_{d \not\succ} \langle\, d \,\rangle \langle dm \setminus \{\, d \,\} \rangle \varphi$$

where $d \not\succ$ quantifies over all minimal deltas in dm (Notation 1.12, page 20; and Definition 1.23, page 22).

Proof: Induction on the size of D. □

▶ **6.9. Corollary:** For nonempty delta model $dm = (D, \prec)$ and all formulas φ:

$$\Vdash_{\Delta\cup\mathsf{F}} \quad [\,(\varnothing, \varnothing)\,] \varphi \leftrightarrow \varphi$$
$$\Vdash_{\Delta\cup\mathsf{F}} \quad [dm] \varphi \leftrightarrow \bigwedge_{d \not\succ} [\,d\,]\, [\,dm \setminus \{\, d \,\}\,] \varphi$$

by taking the inverse of Theorem 6.8. □

[1] The original paper [3], which did not consider conjunctive semantics, presented $\mathbf{K\Delta\cup}$ under the name $\mathbf{K\Delta}$.

[2] From the available relation algebra operators (Section 2.6), the original paper [3] included only axioms for composition $\cdot$ and choice $\sqcup$. We have added axioms for the other operators fundamental to this thesis. This does not include negation $^{-}$, the full delta $\intercal$ or converse $^{\smile}$. Defining converse as a modal operator is rather involved [74], though interesting, and deserves more than a hasty treatment in a small part of this chapter. The other two are simply not that important for us, as well as not constructive (Section 2.6.2).

It is worthwhile to note that the above theorem and corollary are similar to what is known as the *expansion law* of the process algebra CCS [136]. The fact that it works explains why a delta model under disjunctive semantics can be applied to a product simply by applying its deltas in an arbitrary order compatible with $\prec$. A similar law does *not* exist for conjunctive semantics.

Next, we'll use the delta logics as proof systems by looking at their provability relations $\vdash_{\mathbf{K}\boldsymbol{\Delta}_\cup}$ and $\vdash_{\mathbf{K}\boldsymbol{\Delta}_\cap}$ (Definition 1.44), and prove their completeness.

6.3.3 Completeness

It is not hard to see that the delta logics are sound with respect to their respective frames (Definition 1.45, page 27). More interesting is the issue of their completeness (Definition 1.46). It turns out they are strongly complete. Except for the delta model axioms, the presented logic is a subset of dynamic logic [69]; one without iteration. Because there is no iteration axiom, and because delta models are finite and do not contain cycles, modalities can be completely reduced to simple deltas. We define a translation function 'kt':

▶ **6.10. Definition:** Given a set of simple deltas $\mathcal{D}$ and a set of propositional variables *PROP*, we define a translation function $\mathrm{kt}\colon \Psi \to \Psi$ such that for all propositional variables $k \in \mathit{PROP}$, all simple deltas $d \in \mathcal{D}$, all delta models $dm \in \mathcal{DM}_{\mathcal{D}_\triangle}$, all delta (model) expressions $x, y \in \mathcal{D}_\triangle$ and all formulas $\varphi, \psi \in \Psi$:

$$
\begin{array}{lcl}
\mathrm{kt}(\, k \,) & \stackrel{\text{def}}{=} & k \\
\mathrm{kt}(\, \neg\varphi \,) & \stackrel{\text{def}}{=} & \neg\mathrm{kt}(\, \varphi \,) \\
\mathrm{kt}(\, \varphi \vee \psi \,) & \stackrel{\text{def}}{=} & \mathrm{kt}(\, \varphi \,) \vee \mathrm{kt}(\, \psi \,) \\
\mathrm{kt}(\, \langle y \cdot x \rangle\varphi \,) & \stackrel{\text{def}}{=} & \mathrm{kt}(\, \langle x \rangle\langle y \rangle\varphi \,) \\
\mathrm{kt}(\, \langle x \sqcup y \rangle\varphi \,) & \stackrel{\text{def}}{=} & \mathrm{kt}(\, \langle x \rangle\varphi \vee \langle y \rangle\varphi \,) \\
\mathrm{kt}(\, \langle x \sqcap y \rangle\varphi \,) & \stackrel{\text{def}}{=} & \mathrm{kt}(\, \langle x \rangle\varphi \wedge \langle y \rangle\varphi \,) \\
\mathrm{kt}(\, \langle \varepsilon \rangle\varphi \,) & \stackrel{\text{def}}{=} & \mathrm{kt}(\, \varphi \,) \\
\mathrm{kt}(\, \langle \bot \rangle\varphi \,) & \stackrel{\text{def}}{=} & \bot \\
\mathrm{kt}(\, \langle\, d \,\rangle\, \varphi \,) & \stackrel{\text{def}}{=} & \langle\, d \,\rangle\, \mathrm{kt}(\, \varphi \,)
\end{array}
$$

With one of the following depending on delta model semantics:

$$
\begin{array}{lcll}
\mathrm{kt}(\, \langle dm \rangle\varphi \,) & \stackrel{\text{def}}{=} & \bigvee_{d \in \mathrm{derv}(dm)} \langle\, d \,\rangle\, \varphi & \text{(disjunctive semantics)} \\
\mathrm{kt}(\, \langle dm \rangle\varphi \,) & \stackrel{\text{def}}{=} & \bigwedge_{d \in \mathrm{derv}(dm)} \langle\, d \,\rangle\, \varphi & \text{(conjunctive semantics)}
\end{array}
$$ ⌟

The idea behind this function is to translate any formula into an equivalent formula in which all unary modalities are labeled only by simple deltas. This enables us to forget about arbitrary algebraic expressions and delta models, and to construct our completeness proof in terms of the completeness of **K** with regard to the class of all frames (Theorem 1.47, page 27).

▶ **6.11. Lemma:** For all sets of formulas Γ and all formulas φ, we have:

a. $\Gamma \vdash_{\mathbf{K}\boldsymbol{\Delta}\cup} \varphi \iff \Gamma \vdash_{\mathbf{K}\boldsymbol{\Delta}\cup} \mathrm{kt}(\varphi)$

b. $\Gamma \Vdash_{\Delta\cup\mathsf{F}} \varphi \iff \Gamma \Vdash_{\Delta\cup\mathsf{F}} \mathrm{kt}(\varphi)$

c. $\Gamma \Vdash_{\Delta\cup\mathsf{F}} \mathrm{kt}(\varphi) \iff \Gamma \Vdash \mathrm{kt}(\varphi)$

as well as the same for $\mathbf{K}\boldsymbol{\Delta}\cap$ and $\Delta\cap\mathsf{F}$.

Proof: (a) and (b) can be proved by induction (on the complexity of formulas as well as that of delta terms); (c) follows from the observation that for any translated formula, only the relations corresponding to simple deltas are used: hence, we are simply treating our delta frame as a regular frame. □

▶ **6.12. Theorem:** $\mathbf{K}\boldsymbol{\Delta}\cup$ (resp. $\mathbf{K}\boldsymbol{\Delta}\cap$) is strongly complete w.r.t. the class of delta kripke frames $\Delta\cup\mathsf{F}$ (resp. $\Delta\cap\mathsf{F}$).

Proof: This amounts to saying that, for any Γ and φ, if $\Gamma \Vdash_{\Delta\cup\mathsf{F}} \varphi$, then $\Gamma \vdash_{\mathbf{K}\boldsymbol{\Delta}\cup} \varphi$. If $\Gamma \Vdash_{\Delta\cup\mathsf{F}} \varphi$ then, by Lemma 6.11b, we have $\Gamma \Vdash_{\Delta\cup\mathsf{F}} \mathrm{kt}(\varphi)$ and by Lemma 6.11c, we have $\Gamma \Vdash \mathrm{kt}(\varphi)$. Completeness of $\mathbf{K}$ now gives $\Gamma \vdash_{\mathbf{K}} \mathrm{kt}(\varphi)$ and, because $\mathbf{K} \subseteq \mathbf{K}\boldsymbol{\Delta}\cup$, we also get $\Gamma \vdash_{\mathbf{K}\boldsymbol{\Delta}\cup} \mathrm{kt}(\varphi)$. Finally, Lemma 6.11a yields $\Gamma \vdash_{\mathbf{K}\boldsymbol{\Delta}\cup} \varphi$. □

It is possible to extend this completeness result in simple and straightforward ways, because any formula in $\mathbf{K}$ yields a complete axiomatization for the class of frames it defines [42].

6.13. Example: Consider the class of deterministic deltoid Kripke frames $\Delta\cup\mathsf{dF}$ (resp. $\Delta\cap\mathsf{dF}$), in which all simple deltas are deterministic (Definition 2.26). This class of frames can be characterized by the following axiom schema. For all simple deltas d and formulas φ:

$$\langle d \rangle \varphi \to [d] \varphi$$

We call the delta logic generated by that axiom schema $\mathbf{K}\boldsymbol{\Delta}\cup\mathbf{d}$ (resp. $\mathbf{K}\boldsymbol{\Delta}\cap\mathbf{d}$).⌟

6.14. Theorem: The logic $\mathbf{K}\boldsymbol{\Delta}\cup\mathbf{d}$ (resp. $\mathbf{K}\boldsymbol{\Delta}\cap\mathbf{d}$) is strongly complete with regard to the class of deterministic delta frames $\Delta\cup\mathsf{dF}$ (resp. $\Delta\cap\mathsf{dF}$). □

6.3.4 Delta Contracts

Our modal product formulas are essentially syntactic representations of product sets. But they also allow us to syntactically characterize deltas based on their effect on such product sets. We can formulate *delta contracts* reminiscent of Hoare triples:

▶ **6.15. Definition (Delta Contracts):** A *delta contract* is a pair of product formulas $(\varphi, \psi) \in \Psi \times \Psi$, where φ is the *precondition* and ψ is the *postcondition*. ⌟

The following is a way to prove, in a fully syntactic manner, that a specific delta satisfies a specific delta contract:

▶ **6.16. Definition (Contract Provability):** A given delta $d \in \mathcal{D}_{\triangle}$ is *provably correct* with regard to delta contract $(\varphi, \psi) \in \Psi \times \Psi$ iff the following holds:

$$\begin{array}{lcll} d \vdash (\varphi,\psi) & \overset{\text{def.}}{\iff} & \varphi \vdash_{\mathbf{K\Delta\cup}} & [\,d\,]\,\psi \\ d \vdash_{\text{tot}} (\varphi,\psi) & \overset{\text{def.}}{\iff} & \varphi \vdash_{\mathbf{K\Delta\cup}} \langle\,d\,\rangle\top \wedge [\,d\,]\,\psi & \\ & & \underbrace{}_{\text{a}} \quad \underbrace{}_{\text{b}} \quad \underbrace{}_{\text{c}} & \end{array}$$

An analogous definition can be given for conjunctive semantics. ⌟

Basically, a delta d is said to be provably correct with regard to a contract (φ, ψ) iff (a) given a product satisfying the premise φ, (b) delta d is applicable to that product (for total correctness), and (c) all products resulting from the application of delta d satisfy ψ.

This simple proof system is sound and complete in the following sense:

▶ **6.17. Theorem:** The way of using the $\mathbf{K\Delta\cup}$ proof system from Definition 6.16 is sound and complete —with regard to all delta frames— in the following sense:

$$\begin{array}{lcl} d \vdash (\varphi,\psi) & \iff & d \vDash [\![\,\varphi\,]\!] \times [\![\,\psi\,]\!] \\ d \vdash_{\text{tot}} (\varphi,\psi) & \iff & d \vDash_{\text{tot}} [\![\,\varphi\,]\!] \times [\![\,\psi\,]\!] \end{array}$$

where $\vDash$ and $\vDash_{\text{tot}}$ represent delta correctness (Definition 2.29, page 45).

Proof: The following proves the total correctness version:

$$\begin{array}{lll} & d \vdash_{\text{tot}} (\varphi,\psi) & \\ \overset{1}{\iff} & \varphi \vdash_{\mathbf{K\Delta\cup}} \langle\,d\,\rangle\top \wedge [\,d\,]\,\psi & \\ \overset{2}{\iff} & \varphi \Vdash_{\Delta\cup\mathsf{F}} \langle\,d\,\rangle\top \wedge [\,d\,]\,\psi & \\ \overset{3}{\iff} & \forall p \in \mathcal{P}\colon\ (\,p \Vdash \varphi\,) \implies (\,p \Vdash \langle\,d\,\rangle\top \wedge [\,d\,]\,\psi\,) & \\ \overset{4}{\iff} & \forall p \in \mathcal{P}\colon\ p \in [\![\,\varphi\,]\!] \implies p \in [\![\,\langle\,d\,\rangle\top \wedge [\,d\,]\,\psi\,]\!] & \\ \overset{5}{\iff} & \forall p \in \mathcal{P}\colon\ p \in [\![\,\varphi\,]\!] \implies p \in [\![\,\langle\,d\,\rangle\top]\!] \cap [\![\,[\,d\,]\,\psi\,]\!] & \\ \overset{6}{\iff} & \forall p \in \mathcal{P}\colon\ p \in [\![\,\varphi\,]\!] \implies (\,p \in \mathrm{pre}\,[\![\,d\,]\!]\,) \wedge (\,[\![\,d\,]\!](p) \subseteq [\![\,\psi\,]\!]\,) & \\ \overset{7}{\iff} & \forall p \in [\![\,\varphi\,]\!]\colon\ \varnothing \subset [\![\,d\,]\!](p) \subseteq [\![\,\psi\,]\!] & \\ \overset{8}{\iff} & d \in (\,[\![\,\varphi\,]\!] \Mapsto_{\text{tot}} [\![\,\psi\,]\!]\,) & \\ \overset{9}{\iff} & d \vDash_{\text{tot}} [\![\,\varphi\,]\!] \times [\![\,\psi\,]\!] & \end{array}$$

Step 1 applies Definition 6.16. Step 2 applies the completeness result of Theorem 6.12. Step 3 applies Definition 1.42 (page 26) of local consequence. Step 4 twice applies Lemma 6.6. Steps 5 and 6 apply Definition 6.5 (though some steps are skipped). Step 7 applies a number of general simplifications.

Steps 8 and 9 apply Definition 2.31 and Lemma 2.32, confirming what the reader possibly already suspected after seeing the use of the Cartesian product in the theorem: delta contracts are syntactic representations of *delta derivations* (Section 2.4.3). □

This tells us something about the power of the delta logics presented in this chapter. Though they give us valuable insight into the behavior of deltas, they are actually rather limited when it comes specifying the behavior of 'practical'

deltas. For example, while they would be able to specify that software delta (**remove class** C) only accepts products that have a class C, and that it is guaranteed to yield a product without such a class, they are unable to express that the delta leaves all other artefacts the way they are.

One way to express this would be to use a modal language that can refer back to the original world in the frame: a *hybrid language* [22, 40]. This is presented as future work in Chapter 9.

6.4 Kripke Models

As we can now reason on the frame level with the proof system of Section 6.3, we would also like to reason on the level of models.

Recall that a Kripke model is a Kripke frame augmented with a valuation function, which maps propositional variables to the set of worlds in which they are true. Our worlds are products from $\mathcal{P}$. What we'd actually like to reason about is the *features* that are implemented by those products; or more accurately, the *feature combinations*. We want to prove properties about the effects deltas can have on products that satisfy specific feature combinations. So we state that $\mathrm{Pow}(\mathcal{F}) \subseteq \mathit{PROP}$. This is in line with Definition 4.17 on page 107, where it is also explained why we need to handle feature combinations explicitly: it is possible to implement multiple features without implementing their combination.

6.4.1 Proof System Soundness

The ultimate goal here is to formulate some axioms about specific features (i.e., propositional variables in a Kripke model), and then to prove properties about the effects of deltas on those features. However, the proof system for the frame level Definition 1.44 is not sound with respect to global semantic entailment on models. For example, consider the following 'proof':

$$\begin{array}{lll} (1) & F \to \langle\, d \,\rangle\, G & \text{axiom} \\ (2) & F \to \langle\, d \,\rangle\, \neg G & \text{uniform substitution on } G \end{array}$$

So we have $F \to \langle\, d \,\rangle\, G \vdash_{\mathbf{K}\boldsymbol{\Delta}^{\cup}} F \to \langle\, d \,\rangle\, \neg G$, but at the same time the (global) semantic consequence

$$F \to \langle\, d \,\rangle\, G \Vdash^{g}_{\Delta \cup \mathsf{F}} F \to \langle\, d \,\rangle\, \neg G$$

is easily seen to be false. The culprit is our use of uniform substitution. The initial axiom in our false proof is not meant to be a tautology that is "true for all G". It is meant as a statement about the feature G specifically. We can't take away the uniform substitution rule, however. We still need it to prove such truths as:

$$\begin{array}{lll} (1) & k \vee \neg k & \text{propositional tautology} \\ (2) & [\, d \,]\, F \vee \neg [\, d \,]\, F & \text{uniform substitution on } k \end{array}$$

The trick is to allow uniform substitution only on newly produced proposition-letters, but not on the original features in our axioms. This is accomplished by first transforming all propositions in our axioms and formulas to *nullary modalities* [41], on which uniform substitution does not apply. We can then prove valid formulas in the system of frames.

So we now introduce nullary modalities, which may be seen as propositional constants, into the modal language (Definition 1.36). A nullary modality labeled with a propositional variable k is denoted $\textcircled{k}$. This extend frames with a set of predicates on worlds. A nullary modality $\textcircled{k}$ corresponds to a predicate P_k in a frame:

▶ **6.18. Definition (Nullary Modality Semantics):** Given a Kripke frame $\mathfrak{F} = (W, M, \mathrm{U}, \mathrm{R})$ —which now includes a function $\mathrm{U}\colon \mathit{PROP} \to \mathrm{Pow}(W)$, mapping each propositional variable $k \in \mathit{PROP}$ to a corresponding predicate $\mathsf{P}_k \subseteq W$—, a nullary modality $\textcircled{k}$ has the following semantics:

$$\mathfrak{M}, p \Vdash \textcircled{k} \quad \overset{\text{def.}}{\Longleftrightarrow} \quad p \in \mathrm{U}(k)$$

Or equivalently, as an extension to Definition 6.5:

$$[\![\textcircled{k}]\!] \quad \overset{\text{def}}{=} \quad \mathrm{U}(k) \qquad \lrcorner$$

We define the following function to translate propositional variables to corresponding nullary modalities:

▶ **6.19. Definition:** Define the function $\mathrm{u}\colon \Psi \to \Psi$, which transforms all propositional variables in a formula into nullary modalities. For all propositional variables $k \in \mathit{PROP}$ and all formulas $\varphi \in \Psi$:

$$\begin{aligned} \mathrm{u}(k) \quad &\overset{\text{def}}{=} \quad \textcircled{k} \\ \mathrm{u}(\neg\varphi) \quad &\overset{\text{def}}{=} \quad \neg\mathrm{u}(\varphi) \\ &\vdots \end{aligned}$$

For the other shapes of formulas the 'u' translation is simply propagated down to the propositional variables, leaving everything else unchanged. We also lift the function 'u' to sets of formulas in the expected manner. $\lrcorner$

We extend this function to translate from models to frames (overloading the name 'u'):

▶ **6.20. Definition:** Extend translation function 'u' to take a model $\mathfrak{M} = (W, M, \mathrm{R}, \mathrm{V})$ and return a frame:

$$\mathrm{u}(\mathfrak{M}) \quad \overset{\text{def}}{=} \quad (W, M, \mathrm{U}, \mathrm{R})$$

Where U maps propositional variables to the set of worlds in which they are true. So essentially we take $\mathrm{U} \overset{\text{def}}{=} \mathrm{V}$. $\lrcorner$

The following translation lemma holds:

▶ **6.21. Lemma:** For all models $\mathfrak{M}$, worlds w and sets of formulas Γ, we have:

a. $\mathfrak{M}, w \Vdash \Gamma \iff \mathrm{u}(\mathfrak{M}), w \Vdash \mathrm{u}(\Gamma)$

b. $\mathfrak{M} \Vdash \Gamma \iff \mathrm{u}(\mathfrak{M}) \Vdash \mathrm{u}(\Gamma)$

Proof: Proof of (a) is by induction on the complexity of (sets of) formulas. The base case trivially follows from our construction of nullary modalities in terms of propositional variables. (b) follows trivially from (a). □

This lemma enables us to prove the following soundness result with regard to global truth on the model level:

▶ **6.22. Theorem:** For all sets of formulas Γ and all formulas φ:

$$\mathrm{u}(\Gamma) \vdash \mathrm{u}(\varphi) \implies \Gamma \Vdash^{g} \varphi$$

Proof: Assume $\mathrm{u}(\Gamma) \vdash \mathrm{u}(\varphi)$. Let $\mathfrak{M}$ be a model (based on a delta frame) such that $\mathfrak{M} \Vdash \Gamma$. Then, by Lemma 6.21b, we have $\mathrm{u}(\mathfrak{M}) \Vdash \mathrm{u}(\Gamma)$. Now let Λ be the logic of the class of delta frames

$$\{\, \mathfrak{F} \mid \mathfrak{F} \Vdash \mathrm{u}(\Gamma) \,\}.$$

Because Λ is a normal modal logic, it is closed under proof rules, and hence it follows from $\mathrm{u}(\Gamma) \vdash \mathrm{u}(\varphi)$ combined with the fact that $\mathrm{u}(\Gamma) \subseteq \Lambda$, that $\mathrm{u}(\varphi) \in \Lambda$. It follows that $\mathrm{u}(\varphi)$ is valid on this class of frames, so we have:

$$\mathrm{u}(\mathfrak{M}) \Vdash \mathrm{u}(\varphi).$$

Lemma 6.21b now gives us $\mathfrak{M} \Vdash \varphi$ and hence $\Gamma \Vdash^{g} \varphi$. □

Note that this result is valid for all normal modal logics and corresponding frames. It is not specific to delta logics. But we'll now demonstrate it by proving a delta modeling result.

6.4.2 Example

We now illustrate the use of $\mathbf{K\Delta\cup}$ through an example proof. Say we have the feature model as shown in Figure 6.2. The features F, G and H are implemented by the delta model dm in Figure 6.3. The feature T is satisfied in some empty core product, on which we'd like to apply those deltas.

We now introduce a set of basic axioms valid in this model:

6.23. Axiom (Delta Model Axioms): The following are assumed to hold:

(1)	F	$\leftrightarrow$	T	(6)	T	$\leftrightarrow$	$[w]\,F$
(2)	G	$\leftrightarrow$	F	(7)	F	$\leftrightarrow$	$[x]\,G$
(3)	H	$\leftrightarrow$	F	(8)	F	$\leftrightarrow$	$[y]\,H$
(4)	G	$\leftrightarrow$	$[y]\,G$	(9)	G	$\leftrightarrow$	$[z]\,G$
(5)	H	$\leftrightarrow$	$[x]\,H$	(10)	H	$\leftrightarrow$	$[z]\,H$

⌟

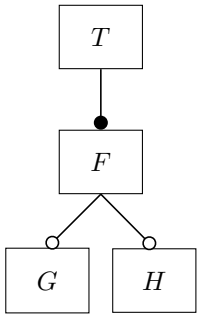


Figure 6.2: Example feature model

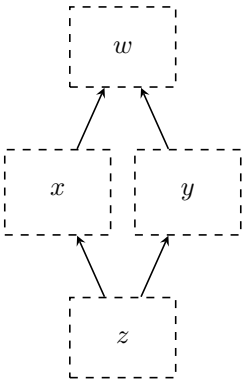


Figure 6.3: Example delta model dm

Axioms (1), (2) and (3) are due to the feature model shown in Figure 6.2. It is generally the case that when a subfeature is implemented its superfeature is implemented as well. (4) and (5) are due to a property we assume the underlying deltoid to have, called non-interference [5], which states that commuting deltas cannot interfere with each others features. (6) to (10) are by design of the deltas: they were developed such that w, x and y implement the features F, G and H (6, 7 and 8), taking into account only the deltas 'above' them, and that conflict resolving delta z does not break the features implemented by the previous deltas (9 and 10).

Axioms (6) to (10) are enforced by the developers of the product line if they follow the workflow to be described in Chapter 7. It ensures desirable global properties by design if local constraints such as axioms (6) to (10) are met. Now say we have a core product $c \in \mathcal{P}$ with $c \Vdash T$. For our example, we'd like to prove the following global property about delta model dm:

6.24. Lemma: $c \Vdash [dm](T \wedge F \wedge G \wedge H)$

In order to prove this property more succinctly, we introduce the following auxiliary proof rules:

6.25. Lemma: For all formulas φ, ψ and χ, and for all box modalities $[d_1], ..., [d_n]$, we have:

$$\varphi \to [d_1] \cdots [d_n]\psi, \quad \psi \to \chi \quad \vdash \quad \varphi \to [d_1] \cdots [d_n]\chi$$

Proof: By induction on n. □

6.26. Lemma: For all formulas φ and ψ and all box modalities $[\,d\,]$, we have:

$$\vdash \quad ([\,d\,]\varphi \wedge [\,d\,]\psi) \;\leftrightarrow\; [\,d\,](\varphi \wedge \psi)$$

Proof: See [42, Example 1.40]. □

The numbers 1 to 10 in the proof of Lemma 6.24 refer to the 'u' translation of the corresponding item from Axiom 6.23.

Proof of Lemma 6.24:

(11)	$\textcircled{T} \leftrightarrow [w][x]\textcircled{G}$	lem 6.25: 6, 7
(12)	$\textcircled{T} \leftrightarrow [w][x](\textcircled{F} \wedge \textcircled{G})$	lem 6.25: 11, 2
(13)	$\textcircled{T} \leftrightarrow [w][x](\textcircled{F} \wedge \textcircled{G} \wedge [y]\textcircled{H})$	lem 6.25: 12, 8
(14)	$\textcircled{T} \leftrightarrow [w][x](\textcircled{G} \wedge [y]\textcircled{H})$	lem 6.25: 13, 2
(15)	$\textcircled{T} \leftrightarrow [w][x]([y]\textcircled{G} \wedge [y]\textcircled{H})$	lem 6.25: 14, 4
(16)	$\textcircled{T} \leftrightarrow [w][x][y](\textcircled{G} \wedge \textcircled{H})$	lem 6.26: 15
(17)	$\textcircled{T} \leftrightarrow [w][x][y]([z]\textcircled{G} \wedge \textcircled{H})$	lem 6.25: 16, 9
(18)	$\textcircled{T} \leftrightarrow [w][x][y]([z]\textcircled{G} \wedge [z]\textcircled{H})$	lem 6.25: 17,10
(19)	$\textcircled{T} \leftrightarrow [w][x][y][z](\textcircled{G} \wedge \textcircled{H})$	lem 6.26: 18
(20)	$\textcircled{T} \leftrightarrow [w][x][y][z](\textcircled{F} \wedge \textcircled{G} \wedge \textcircled{H})$	lem 6.25: 19, 2
(21)	$\textcircled{T} \leftrightarrow [w][x][y][z](\textcircled{T} \wedge \textcircled{F} \wedge \textcircled{G} \wedge \textcircled{H})$	lem 6.25: 20, 1
(22)	$\textcircled{T} \leftrightarrow [w][x][y][dm_1](\textcircled{T} \wedge \textcircled{F} \wedge \textcircled{G} \wedge \textcircled{H})$	lem 6.25: 21, $\Delta\forall$
(23)	$\textcircled{T} \leftrightarrow [w][x][dm_2](\textcircled{T} \wedge \textcircled{F} \wedge \textcircled{G} \wedge \textcircled{H})$	lem 6.25: 22, $\Delta\forall$

Formula (24) is derived in a manner symmetric to formula (23).

(24)	$\textcircled{T} \leftrightarrow [w][y][dm_3](\textcircled{T} \wedge \textcircled{F} \wedge \textcircled{G} \wedge \textcircled{H})$	symmetric
(25)	$\textcircled{T} \leftrightarrow [w][x][dm_2](\textcircled{T} \wedge \textcircled{F} \wedge \textcircled{G} \wedge \textcircled{H})$ $\wedge\, [w][y][dm_3](\textcircled{T} \wedge \textcircled{F} \wedge \textcircled{G} \wedge \textcircled{H})$	$I_\wedge$: 23,24
(26)	$\textcircled{T} \leftrightarrow [w]([x][dm_2](\textcircled{T} \wedge \textcircled{F} \wedge \textcircled{G} \wedge \textcircled{H})$ $\wedge\, [y][dm_3](\textcircled{T} \wedge \textcircled{F} \wedge \textcircled{G} \wedge \textcircled{H}))$	lem 6.26: 25
(27)	$\textcircled{T} \leftrightarrow [w][dm_4](\textcircled{T} \wedge \textcircled{F} \wedge \textcircled{G} \wedge \textcircled{H})$	lem 6.25: 26, $\Delta\forall$
(28)	$\textcircled{T} \leftrightarrow [dm](\textcircled{T} \wedge \textcircled{F} \wedge \textcircled{G} \wedge \textcircled{H})$	lem 6.25: 27, $\Delta\forall$

where

$$dm_1 = dm \setminus \{w, x, y\} \qquad dm_3 = dm \setminus \{w, y\}$$
$$dm_2 = dm \setminus \{w, x\} \qquad dm_4 = dm \setminus \{w\}$$

Then, by $c \Vdash \textcircled{T}$, we have our result. □

Many steps are skipped in this proof, mostly those concerned with invoking propositional tautologies and applying modus ponens. We have kept only the more interesting steps — those that directly use our axioms.

Since satisfiability for the normal multimodal logic is decidable (in fact, it is PSPACE-complete [41]), and the special modal operators of delta logic can be trivially translated away (Definition 6.10), proofs such as this one can be automated.

6.5 Conclusion

Much of ADM is dedicated to the goal of developing syntactic languages and techniques for semantic concepts. Deltas are syntactic. But products (from the ADM point of view) are semantic concepts. Consequently, reasoning about the semantics of deltas requires semantic proof machinery.

This chapter describes how modal logic can solve this problem. Given any kind of decidable specification language for the product domain, wrapping a multi-modal logic around it enables us to prove that certain deltas implement certain features, that they do not break existing features, and so on. The result is a language reminiscent of dynamic logic, but lacking a construct for iteration, making the logic decidable.

The chapter shows that the modal proof system can be used to prove certain kinds of delta correctness, but in Section 6.3.4 we discover that it cannot be used in delta postconditions to refer back to the original product. They are therefore unable, for instance, to specify that software delta (**`remove class`** `C`) does not modify any classes other than `C`. Chapter 9 briefly discusses how an extension to hybrid logic [22, 40] may be used to overcome this without losing decidability.

6.6 Related Work

Completeness proofs in modal logic have a long-standing history, closely tied to the history of relational semantics based on Kripke frames. A comprehensive survey of this history can be found in e.g. [42, Section 1.8].

The modal logic presented in this chapter has a flavour very reminiscent of dynamic logics such as PDL [42, 69]. A crucial difference, however, is that the logic presented here is simpler (and hence, easier to work with) due to the absence of iteration. Due to this simplicity, complex modalities can be easily unraveled into simpler ones, enabling the main results from Sections 6.3 and 6.4.

Partial motivation for the presented delta logics is to make formal properties in the Delta Modeling Workflow (Chapter 7) more transparent. The proof of Lemma 6.24 is just an example of a proof of product line completeness for a specific case.

Finally, it is worth noting that the typesystem described by Lienhardt and Clarke [120], unlike the logic of this chapter, is able to specify that deltas in an object oriented setting do not modify unmentioned artefacts, by regarding them as polymorphic functions.