Chapter 15 – Twins at Scale: Ecosystems

15.0 Before you start

Where we are. Twelve chapters defer something to this one, and every one of them defers the same thing: what happens when there is more than one twin. Chapter 3 said "composing many twins – Chapter 15". Chapter 9 asked what happens when 400 substations become 400 twins. Chapter 11 said its alert arithmetic "is the part that scales worst". Chapter 12 said this is where its `N` "stops being a parameter and becomes the subject". Chapter 13 said the counterparty count does the same. Chapter 14 said this chapter takes every number in its operating bill and multiplies it.

Four separate chapters have each claimed that something of theirs scales worst. Chapter 10 said the context model does not scale by copying, Chapter 11 said the signal ratio, Chapter 14 said announced triggers and, separately, purpose-drift detection. They cannot all be right, and ranking them is a job only the last chapter can do.

The register, unchanged from Chapters 9 to 14. You have scaled a system before. You know the interesting question is which quantity grows fastest, not how to add machines.

> **This chapter does not teach scaling. It performs a scaling analysis of this book's own results** – every quantity Parts II and III computed, asked how it behaves in `N` – **and finds that two of them grow faster than the fleet does.**

And it has a second job. This is the last chapter. Sec. 15.9 closes the book.

What you are assumed to know. Everything. This chapter reuses more earlier results than any other and computes almost nothing new: Chapter 1's value metric, Chapter 7's calibration campaigns and credibility argument, Chapter 10's context model and dataset definitions, Chapter 11's alert economics, Chapter 12's cost model, Chapter 13's counterparty rule, and **Chapter 14's operating bill, which is the number everything here multiplies.**

The maths budget. As Chapters 9 to 14. Three arithmetic set pieces, all of them reusing earlier numbers: a unit conversion that has to come first (Sec. 15.1.3), the commissioning cost that actually breaks (Sec. 15.2.5), and the mesh case Chapter 13 deferred (Sec. 15.4.1).

What this chapter deliberately does not cover. Distributed-systems engineering – no sharding, no consensus, no partitioning; what is different about a fleet of twins is which *quantities* grow, not how to add machines. An ecosystem survey: the corpus offers composite, federated, integrated, system-of-systems and collaborative framings, and Sec. 15.4.2 notes the proliferation rather than cataloguing it. Data spaces in depth – one bounded subsection. Federated learning as a technology – Chapter 8 named it, Chapter 12 placed it. Governance and organisational design, named where the sources name them. **And nothing new about a single twin**: everything here is a consequence of `N`.

Learning objectives

By the end of this chapter, you will be able to:

1. **Decide** how many twins a fleet of assets should be, and **state** the cost consequence of the boundary you chose.
2. **Sort** a twin's costs into what stays fixed, what grows with the fleet and what grows faster, and **identify** which quantity binds first.
3. **Convert** a per-asset operating practice into a population practice – sampled validation, ranking, population calibration – and **state** what each trades away.
4. **Explain** what changes about a credibility argument when a twin's inputs are other twins, and **apply** Chapter 13's counterparty rule in the mesh case.
5. **Recognise** which problems at this scale are genuinely unsolved, and say so.

15.1 How many twins is this?

15.1.1 The question nobody asks first

Four hundred substations. Eighty turbines. Twelve pots. **How many twins?**

The question sounds like bookkeeping and it is the most consequential decision in this chapter, because every quantity in Sec. 15.2 is measured *per twin* and the boundary is a choice.

Notice that this book has already made the choice twice without discussing it. The demonstrator is *one* twin covering twelve pots – Chapter 3 architected one twin, Chapter 7 wrote one credibility argument, Chapter 14 costed one operating bill. The turbine fleet has been described as eighty twins. Nobody argued for either.

15.1.2 Three ways to divide, and what each costs

Division	What it is	Cheap	Expensive
One twin per asset	400 twins, each with its own model, argument and register	Per-asset decisions are natural; assets can differ	Everything in Chapter 14's bill, 400 times
One twin for the whole fleet	One model family, one argument, one register, 400 bindings	Chapter 14's per-twin lines paid once	Per-asset decisions become awkward; one credibility argument must cover 400 different situations
A hierarchy	Asset twins beneath a system twin	Matches how people think about the plant	Two levels to build, two to operate, and the interface between them is Sec. 15.4's problem

Hierarchical composition is a recognised architectural capability rather than an improvisation: the ability to compose two or more digitalised entities into a higher-level entity

"that lives thanks to information flowing from the underlying structures" is one of the named properties such architectures set out to provide [1].

15.1.3 The unit conversion, before any arithmetic

Here is the trap, and it catches people because Chapter 14 handed you a single number.

> **Chapter 14's 17.3 engineer-days a year is per *twin*, and the demonstrator's twin covers twelve pots.**

So "scale to 400 assets" is not "multiply by 400". It is multiply by 400 **only if you have also decided there are 400 twins**, and that is the decision of Sec. 15.1.1 rather than a property of the substations.

Work out what Chapter 14's bill actually contains. Of its seven lines, which depend on the number of *assets* inside the twin, and which do not?

Chapter 14's line	Days	Depends on asset count?
Recalibration and re-validation	6.5	**No** – one refit cycle covers all twelve pots. Grows with the number of *exceptions* to review, not with pots
Alert handling	1.3	**Yes** – more assets, more alerts
Cannot-evaluate digest review	3.0	No – one digest
Sensor recalibration or replacement	1.0	**Yes**
Dependency and platform upgrades	2.0	No
Incident investigation	2.0	Weakly – more assets, more incidents, but shared learning
Recording physical changes	1.5	**Yes**
	17.3	

Roughly 11.5 of the 17.3 days are per-twin, about 4 are per-asset, and 2 grow weakly with asset count, and that split is the whole of Sec. 15.2.

> **Which yields the finding this section exists for, and it is the opposite of most people's instinct: the cheapest way to cover more assets is usually to put them inside one twin, not to make more twins.** Twelve pots cost about the same to operate as one, because the work is per refit-cycle, per digest and per upgrade – not per pot.

The limit on that, stated so nobody over-applies it. More assets in one twin works while the assets are **near-identical** and serve **the same decision**. The moment two assets need different models, different thresholds or different credibility claims, one twin covering both has a credibility argument that must be true of both – and Chapter 7's

intended-use statement cannot name two intended uses. That is the real boundary, and it is about the decisions rather than about the hardware.

15.1.4 The test

Three questions, in order:

1. **Do these assets serve the same decision?** If not, they are different twins regardless of how similar the hardware is.
2. **Are they similar enough that one model family fits, with per-asset parameters?** If yes, one twin with many bindings. If no, separate twins or a hierarchy.
3. **Does anybody need a statement about the whole?** "Is the network healthy" is a different question from "is substation 214 healthy", and if somebody asks it, you need a level above.

For the demonstrator: same decision, near-identical pots, nobody asks about the greenhouse as a whole. One twin, twelve bindings – which is what Chapter 3 built, and Sec. 15.1.3 is why it was right.

15.2 What scales, what does not, and what gets worse

15.2.1 The method

Take every quantity Parts II and III computed and ask how it behaves as the number of assets grows. Three columns: fixed, linear, and worse than linear.

Do this before designing anything. The fixed column tells you what a platform amortises (Chapter 12), the linear column tells you what to automate (Chapter 14 Sec. 14.4.5), and the third column tells you what architecture you actually need.

15.2.2 What stays fixed

Quantity	From	Why it does not grow
The software build	Ch1's 30 engineer-days	Written once for a family of assets
Dependency and platform upgrades	Ch14's 2 days	One stack
Expiry-register *structure*	Ch14 Sec. 14.1.2	One register; rows grow, the review does not
Standards adoption `F`	Ch13 Sec. 13.1.3	Paid once
The cannot-evaluate digest	Ch11 Sec. 11.4.6	One digest, more rows
Model *structure*	Ch4	One family for similar assets

This column is why fleets are possible at all, and it is also Chapter 12's answer arriving: a platform's fixed cost `F` amortises over exactly these lines, which is why Chapter 12 Sec. 12.5.2 found the crossover between the first twin and the second.

15.2.3 What grows with the fleet

Quantity	From	At 400 assets
Ingest volume and storage	Ch9 Sec. 9.8.1, Ch10 Sec. 10.3.5	Linear, and Chapter 9's arithmetic says still small for slow assets
Alerts	Ch11 Sec. 11.4.5	Linear in count
Sensor events, physical changes	Ch14 Sec. 14.4.2	Linear, and every one is an **announced** trigger
Recalibration runs	Ch7 Sec. 7.7.2	Linear if per-asset; Sec. 15.3.5 is the alternative
Context rows	Ch10 Sec. 10.4	Linear in assets

Linear is manageable and it is not free. The rule from Chapter 14 Sec. 14.4.5 applies unchanged: automate the largest line first. What changes at scale is that a line costing half a day a year becomes 200 days, so lines nobody would automate for one twin become the first thing to automate for four hundred.

15.2.4 The two that grow faster than the fleet

First: human attention per asset is not available.

Chapter 11 Sec. 11.11 gave the real figure for the district-heating twin – "roughly forty alerts a week" across 400 substations, or about **2,080 a year**. At Chapter 14's 0.1 engineer-day per alert that is **208 engineer-days a year of triage demand.**

Now count the supply. Chapter 13 Sec. 13.10 gave the utility a team of four with no on-call rotation. If one of them spends a tenth of their time on alerts, that is about **22 days a year**.

```
demand  208 engineer-days
supply   22 engineer-days
ratio   about 9 to 1
```

Figure 15.1 is that ratio drawn, and it is drawn as two lines rather than two numbers because the shape is what makes the conclusion unavoidable.

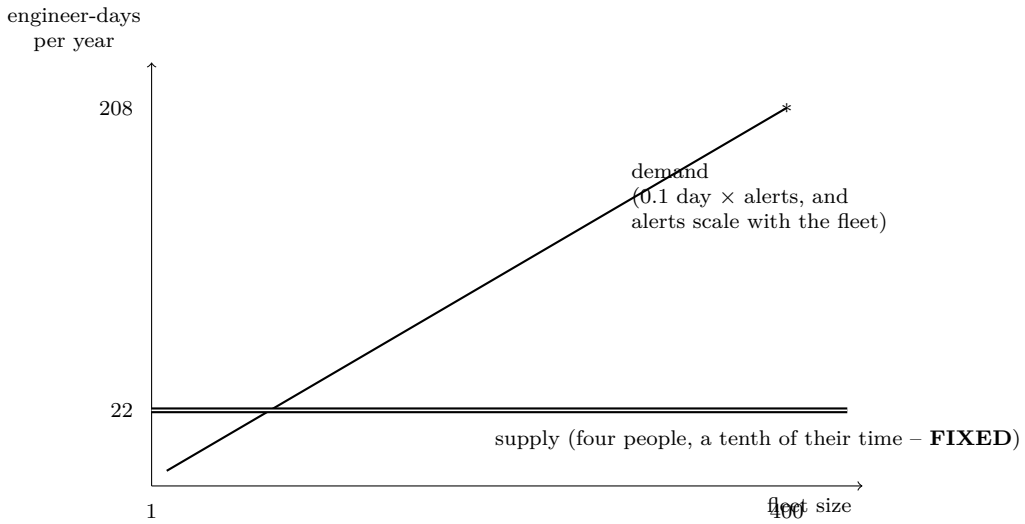


Figure 15.1 Attention does not scale. The gap is not a staffing shortfall to be closed; it is the reason Chapter 11's operators "mostly ignore" the alerts.

> **Which explains a fact this book reported two chapters ago and did not explain.** Chapter 11 Sec. 11.11 said the operators "mostly ignore" the alerts. **They are not being careless. Ignoring roughly eight alerts in nine is the only physically available response**, and no amount of training changes a nine-to-one arithmetic.

This is not linear growth of a cost. It is linear growth of demand against **fixed** supply, which is the same thing as unbounded, and Chapter 11 predicted exactly it: a signal ratio tolerable for one asset is an unusable service for four hundred.

Second: integrations, if twins must talk to each other. `N(N-1)/2` rather than `N`. Sec. 15.4.1 does that arithmetic, because it belongs with the ecosystem material.

And the honourable mention, which is not superlinear and is worse than it looks. Chapter 14 Sec. 14.1.3 found about half of the book's expiry conditions are **announced** – they exist only if a human says something. Those grow linearly in *events*, and the mechanism that detects them – somebody remembering – does not improve with practice or tooling. Chapter 14 Sec. 14.8(a) found the only reliable answer: attach the record to an act the organisation already logs. **At 400 assets that stops being good practice and becomes the difference between a context model that is true and one that is fiction.**

15.2.5 Commissioning, not operating, is what actually breaks

Here is the quantity no chapter costed, and it is larger than everything above.

Chapter 14 costed *operating* a twin. Nobody costed *commissioning* one asset. Look at what Chapter 7 required before a single pot could be trusted: two calibration campaigns (Sec. 7.4.1 and Sec. 7.4.2), a hold-out week (Sec. 7.4.5), a validity envelope built from the assumption ledger (Sec. 7.5.3), an alert threshold derived from the value

metric (Sec. 7.5.2), and a credibility argument (Sec. 7.8.5).

For twelve near-identical pots that was done once. **For 400 substations that are not identical – different vintages, different heat exchangers, different building loads – it is per asset.** Call it 3 engineer-days each, which is generous given that Chapter 7's campaign alone took two weeks of logging:

```
400 assets x 3 engineer-days = 1,200 engineer-days
                             = about 5.5 engineer-years, before anything runs
```

Against Chapter 14's entire annual operating bill of 17.3 days. Commissioning is not a rounding error on operations; at scale it is two orders of magnitude larger, and it is spent before the twin delivers anything.

> **That single figure is why the field has population methods, and why Sec. 15.3 exists.** Every technique in the next section is a way of not paying 1,200 days.

What the alternative costs. Fit a model across the population once, then derive each asset's parameters as an offset from a short window:

```
population fit, once                    about  20 engineer-days
per-asset offset, 400 x 0.25 days       about 100 engineer-days
                                        --------------------------
                                        about 120 engineer-days
```

A factor of ten, and Sec. 15.3.5 is what it costs in accuracy.

15.3 Operating a population instead of many individuals

15.3.1 The shift

Everything in Sec. 15.2's third column has the same answer, and it is a change of unit rather than a change of technique:

> **Stop treating the fleet as `N` twins that each need attention, and start treating it as one population that needs attention.**

Figure 15.3 is the change of unit, with the price of each conversion written against it – because every one of the five is a trade, not a free win.

```
PER ASSET (N twins)          PER POPULATION (one fleet)
-------------------          --------------------------

validate each one     ->     validate a SAMPLE
                             price: a per-asset claim you
                                    can no longer make

alert on each one     ->     RANK them
                             price: something real sits
                                    below the cut line
```

```
context per instance  ->     context as KINDS
                             price: the instance that does
                                    not match its kind

calibrate each one    ->     calibrate across the POPULATION
                             price: the genuine outlier is
                                    pulled toward the mean

watch each purpose    ->     SAMPLE purpose drift
                             price: drift found late on the
                                    assets you did not sample
```

Figure 15.3 Five conversions, five prices. The unit change is the only thing that makes 400 assets tractable, and each row gives something up to buy it.

Five practices follow. Each one converts a per-asset activity into a per-population one, and **each pays for it with something specific** – the chapter's job is to name the price rather than to recommend the practice.

15.3.2 Validation on a sample

Chapter 14 Sec. 14.8(b) already worked this: full re-validation per asset per quarter is eighty days a quarter for eighty turbines, and unaffordable.

The population version. Full re-validation on a rotating sample – say 10 per cent of assets each quarter, so every asset is fully re-validated roughly every two and a half years – plus Chapter 11 Sec. 11.5.4's continuous scoring on **all** of them.

What it costs. An asset outside the sample is covered only by the continuous half, and Chapter 14 Sec. 14.3.3's table says what the continuous half cannot catch: **a model that is wrong in a way the current conditions do not exercise.** So the sampled regime is blind to exactly the seasonal and rare-condition failures that full validation exists to find, for assets not currently sampled.

The mitigation, and it is not more sampling. Promote an asset into the full-validation set when its continuous scores are unusual, when its context says it differs from the population, or when its output feeds a decision that has just become more consequential. **The sample is stratified by risk, not random**, and saying which stratification you used belongs in the credibility argument.

15.3.3 Ranking, not notifying

Sec. 15.2.4 found a nine-to-one gap between alert demand and human supply. No improvement in the detector closes it, because Chapter 11's three fixes were already applied and the remaining alerts are mostly *real*.

The population version. Stop notifying per asset. Produce, once a day, a ranked list of the worst `k` assets – ten, twenty, whatever a person can actually work through – and let a human work down it.

```
attention demand becomes O(1) in N:  k items a day, whatever N is
```

What it costs, precisely. An asset ranked below the cut is not seen today. If it worsens it rises; if it does not, it may never be seen. So:

- **Large faults are safe.** A failed heat exchanger ranks near the top immediately.
- **Small persistent faults become invisible.** An asset sitting at rank 50 with a real but modest problem is never reached.

  That is Chapter 11 Sec. 11.4.7's finding arriving at fleet scale. Confirmation defeated spread and did nothing to bias; **ranking defeats volume and does nothing to the persistent small fault.** The two blind spots compose, and a fleet using both should say so in one sentence of its credibility argument.

The instrument that covers the gap is not another alert. It is Chapter 7 Sec. 7.7.4's residual-bias monitor, run per asset and reviewed as a *distribution* across the population: an asset whose bias has been mildly wrong for six months shows up in a histogram even though it never reached rank ten.

15.3.4 Context as kinds, not instances

Chapter 10 Sec. 10.14 said the context model "is the part that does not scale by copying". Here is why, and what to do.

Why copying fails. Chapter 10's context tables describe *this* pot, *this* sensor, *this* binding. Four hundred copies is four hundred sets of rows to maintain, and every schema change is a migration across all of them. Worse, the interesting relationships at fleet scale are **between** assets – substation 214 is downstream of 187 – and those have no home in a per-asset table.

The population version. Separate what is true of a *kind* from what is true of an *instance*:

Layer	Holds	Rows
Kind	This model of heat exchanger; this sensor type with its range, resolution and drift; this model family with its envelope	Tens
Instance	This substation, its serial numbers, its bindings, its fitted parameters, its validity intervals	Thousands
Relationship	Downstream of, shares a feeder with, same vintage as	Thousands

Three payoffs, each of which is a cost avoided. Chapter 9's six datasheet numbers are stated once per sensor *type*, not per sensor. Chapter 7's envelope is stated once per model *family*. And the relationship layer is what makes Sec. 15.3.5's population possible at all – because "similar enough to share information" is a statement about kinds.

This is where the asset-model literature stops being optional. Chapter 10 Sec. 10.4.6 said a graph becomes worth looking at when the number of *kinds* of relationship starts growing, and named the greenhouse exemplar's knowledge graph, built to record structure and its changes [2]. Four hundred assets with relationships between them is that moment.

15.3.5 Calibration across a population

Sec. 15.2.5 priced per-asset commissioning at 1,200 engineer-days and promised an alternative.

The idea, and it has a literature. Population-based structural health monitoring is exactly "the process of utilising information across a population of structures in order to perform and improve inferences that generalise for the complete population", built on knowledge transfer and mapping between members – and motivated by precisely the fleet situation, where labelled data exists for some members and not others [3]. Probabilistic graphical formulations of twin calibration make the same claim from the other direction: a principled, unified process for calibration and updating that, in the authors' words, is "able to scale to an entire fleet of digital twins" [4].

What it looks like operationally, without any of the mathematics:

1. Pool every asset's data and fit once, so the fit sees the whole population rather than one member of it.
2. Express an individual asset not as its own parameter set but as a **departure** from that pooled result – which is why it needs so much less data than a full per-asset campaign would.
3. Sort by the size of that departure. **A large one means either an asset worth looking at or data worth distrusting**, and a human is needed either way.

What it costs, and this is the part to be honest about. A population model assumes the population is a population. Chapter 7's whole apparatus – the assumption ledger, the envelope, the hold-out – now applies to the *population assumption* as well as to the model. And the failure mode is specific: **an asset that is genuinely unlike its population gets parameters pulled toward the population's, and is therefore modelled worse than it would have been alone** – while looking entirely normal, because its parameters are plausible.

> **So step 3 is not optional bookkeeping, it is the safety mechanism.** The assets whose offsets are unusual are the ones the population method is least entitled to speak about, and reviewing them is what stops a ten-times saving becoming a fleet-wide silent error.

15.3.6 Purpose drift, sampled

Chapter 14 Sec. 14.5.2 found the failure that nothing fires on – a credibility argument that is true and irrelevant – and its only detector was a five-minute question to a user: *what was the last decision you made using this?* Chapter 14 Sec. 14.12 then noted that with one twin you can ask and with four hundred you cannot.

The population version is the obvious one and it works. Ask a sample. Five users a quarter, chosen across roles rather than randomly, is twenty conversations a year and

about two engineer-days.

What it costs. Purpose drift is not uniform across a fleet – it starts somewhere, usually with one enthusiastic user, and spreads. **A sample finds widespread drift reliably and localised drift only by luck.** The compensating instrument is usage data you already have: an asset or a view whose query pattern changed is a candidate for the question, and that turns a random sample into a targeted one for no extra cost.

15.4 Ecosystems: twins that talk to each other

Everything so far is one organisation operating many assets. This section is the other axis: **twins whose inputs are other twins**, frequently across an organisational boundary.

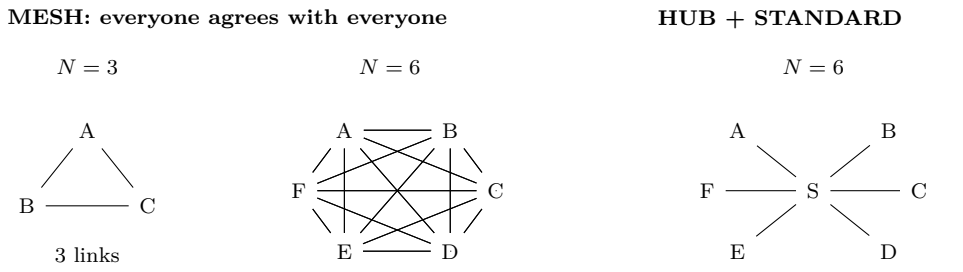
15.4.1 The mesh case, which Chapter 13 deferred

Chapter 13 Sec. 13.1.3 gave the counterparty rule – adopt a standard when `C x I > F` – and Chapter 13's own record of rejected material noted that it used the **hub** case, where you integrate with `C` counterparties who do not integrate with each other, and excluded the full-mesh case as rare for twins. **This is where it stops being rare**, and it arrives on schedule.

The arithmetic. If `N` twins must each exchange with every other, the number of bilateral agreements is `N(N-1)/2`. If instead each adopts a common standard, the number of adoptions is `N`. With Chapter 13's figures – `I` = 4 engineer-days per bilateral integration, `F` = 10 per adoption:

Twins	Bilateral: `N(N-1)/2 x 4`	Standard: `N x 10`	Cheaper
3	12	30	Bilateral
6	60	60	Equal
10	180	100	Standard
400	319,200	4,000	**Standard, by about eighty to one**

Figure 15.2 shows why the table turns over: one shape grows with the square of the fleet and the other grows with the fleet.



N	mesh × 4 days	standard × 10 days
3	12	30
6	60	60 ← break-even
10	180	100
400	319,200	4,000

Figure 15.2 Chapter 13's counterparty rule at fleet scale. The same rule that said "adopt nothing" for one twin says "adopt" past six, and the variable that changed is only how many parties must agree.

Break-even at six mutually-interoperating twins.

> **So Chapter 13's answer for the demonstrator – adopt nothing, because `C = 0` – and this chapter's answer for an ecosystem are the same rule producing opposite results, and the variable is still just how many people you must agree with.** Chapter 13 said that in one line and could not demonstrate it; this table is the demonstration.

One qualification, so the table is not over-applied. Most fleets are *not* meshes. Four hundred substations reporting to one operator is a hub: `N` integrations, not `N(N-1)/2`, and Chapter 13's original arithmetic applies unchanged. **The mesh case needs a reason to exist** – typically that the twins belong to different organisations with no natural centre, which is the situation Sec. 15.4.4's data spaces address.

15.4.2 Composition is not integration

The vocabulary here has proliferated, and it is worth knowing that before reading anything: the same idea appears as composite, federated, integrated, system-of-systems, composable and collaborative digital twins [5]. **Do not spend time on the taxonomy.**

The distinction that does matter is between integration and composition.

Integration is making two systems exchange data. It is a Chapter 9 and Chapter 13 problem, it is well understood, and Sec. 15.4.1 prices it.

Composition is making a twin whose **state and behaviour** derive from other twins – a network twin whose health is a function of 400 substation twins' health. That is a modelling problem, not a plumbing one, and it is where the hard parts live.

Three of the hard parts, each of which the book has the vocabulary for.

Whose time? Chapter 9's four timestamps and Chapter 10's `as_of` were hard enough for one twin. A composite twin's state at time `t` is a function of component states that were each true at slightly different times, with different ages. **Chapter 2's age-of-twin becomes a property of a composition rather than of a stream**, and the composite is at best as fresh as its stalest input.

Whose uncertainty? Chapter 7 reported the demonstrator's prediction as plus or minus 2.0. A composite of 400 such predictions has an uncertainty that depends on whether the component errors are independent – and they usually are not, because the assets share weather, share suppliers and share a calibration procedure. **Assuming independence is the standard error here and it makes the composite look far more precise than it is.**

Whose model? Chapter 4 catalogued model families, and a composite may contain several. Integrating models that were built separately, for separate purposes, is a recognised open problem rather than a solved one [6].

15.4.3 The credibility argument of a composed system

Chapter 7 Sec. 7.13 asked what happens to a credibility argument when twins start composing, and Chapter 7 Sec. 7.7.5 pointed at the answer: an approach in which each side publishes, per trustworthiness characteristic, a link to a human- and machine-readable justification – an assurance case – so a composed system can reason about whether its parts are trustworthy enough for the job [7].

What that means in practice, in three rules.

1. A composite's credibility argument cites its components' arguments; it does not restate them. Chapter 7's five parts still apply at the top level, and the evidence section becomes a list of references plus a statement about the *composition*.

2. The composite's claim is bounded by its weakest component, on every axis separately. Its validity envelope is the intersection of the components' envelopes. Its freshness is the age of the stalest input. Its expiry is the earliest component expiry. **Each of those is a different component**, and computing them is a small, mechanical, and routinely skipped exercise.

3. A component's expiry must reach the composite. Chapter 14's expiry register was one list with one owner; a composed system has a register per twin and **an expiry that fires in a component invalidates the composite silently** unless somebody wired it. That is the single most likely defect in any composed twin, and it is a Chapter 14 artifact extended by one column: *who else depends on this row?*

And the genuinely hard case, stated and not solved. If the components belong to different organisations, you cannot inspect their arguments and must rely on what they publish. That is what the trust-vector proposal is for [7], and the obstacles named around composite twins are exactly the ones you would expect: trust, interoperability, governance, ownership, security and privacy [5]. **Five of those six are not engineering problems**, which is the honest reason ecosystem twins are rarer than the literature's enthusiasm suggests.

15.4.4 Data spaces, named and bounded

Chapters 10, 12 and 13 all deferred data spaces here. One subsection.

What it is, in one sentence. An arrangement – technical and contractual – under which organisations exchange data while each retains control over what is shared, with whom, and on what terms.

Why it exists for twins. Chapter 13's counterparty rule says the value of a standard grows with the number of parties. Data spaces are the same observation applied to the *data* rather than to the interface: they exist because the parties want the exchange and do not want to hand over the asset. Work in this direction covers decentralised approaches [8], asset-oriented exchange [9] and data-driven cross-organisation architectures [10].

The one question that decides whether you need one.

> **Is there data you cannot move, and a party who needs what it would tell them?**

If no data is restricted, ordinary integration is cheaper. If data is restricted and nobody outside needs it, keep it and say nothing. **A data space earns its considerable cost only when both halves are true**, and that is a much rarer situation than the literature's volume suggests. Chapter 8 Sec. 8.3.7 made the identical point about federated learning, and it is the same instinct: architecture adopted for a constraint you do not have is complexity with no purchaser.

15.4.5 What is genuinely unsolved

Chapter 4 Sec. 4.16 parked a reference for this chapter and it is the right one to close on. A review of digital-twin integration, using a smart city as its worked example, names nine integration challenges it considers still outstanding and calls on the community to take them up [6]. That is a 2025 paper describing open problems, not a survey of solved ones.

Four things this chapter has been honest about not solving:

Composing uncertainty correctly (Sec. 15.4.2), where the independence assumption is standard and usually wrong.

Credibility across organisational boundaries (Sec. 15.4.3), where trust vectors are a proposal rather than a practice.

Integrating models built separately for separate purposes [6].

The five non-engineering obstacles – trust, governance, ownership, security, privacy – that a composite twin architecture must address [5], and which no amount of good software resolves.

Why saying this matters more here than anywhere else in the book. Ecosystems are the part of the field with the largest gap between what is published and what is running. A reader who has followed fourteen chapters of "derive it from the decision" should be equipped to notice that gap, and this chapter's last technical act is to name it rather than to close it.

15.5 Worked example: the greenhouse becomes a facility

The lab's twelve-pot bench is a success. The institute funds a facility: **twenty benches, 240 pots, four research groups.**

15.5.1 How many twins?

Sec. 15.1.4's three questions.

Same decision? Mostly. All four groups want watering faults detected. But one group is running a drought-stress experiment in which **failing to water is the intervention**, and a fault detector that alerts on it is worse than useless.

One model family with per-asset parameters? Yes for three groups. The drought group's pots leave Chapter 4's assumption A1 behind entirely.

Anybody asking about the whole? Yes – facility management wants to know water consumption and whether the pump manifold is healthy.

Decision: three twins, not one and not twenty.

Twin	Covers	Why separate
Standard cultivation	15 benches, 180 pots	One decision, one model family, per-pot parameters
Drought experiment	3 benches, 36 pots	**Different decision** – Sec. 15.1.3's real boundary
Facility	Manifold, supply, consumption	Different question, different assets

Notice what decided it. Not the pot count, not the hardware, not the network. **The decisions**, exactly as Sec. 15.1.4 said and exactly as Chapter 1 has said since its second section.

15.5.2 The commissioning bill, and the population answer

Sec. 15.2.5's arithmetic for the standard-cultivation twin:

```
per-pot commissioning at Chapter 7's cost, 180 x 3 days = 540 engineer-days
                                                        = about 2.5 engineer-years
```

Unaffordable for a research facility, and it would have been discovered in month four rather than in planning.

The population version, per Sec. 15.3.5:

```
population fit across 180 pots, once            about 15 engineer-days
per-pot offset from a 3-day window, 180 x 0.1   about 18 engineer-days
review of unusual offsets, say 15 pots x 0.5     about  8 engineer-days
                                                 --------------------
                                                 about 41 engineer-days
```

A factor of thirteen, and the third line is the safety mechanism rather than an overhead: the fifteen pots whose offsets are unusual are where the population assumption is least entitled to speak.

15.5.3 Operating 180 pots

Practice	For 12 pots (Ch14)	For 180
Alerts	13 a year, notify each	**Rank**: a daily top-ten across the facility
Recalibration	13 cycles, all pots	Same 13 cycles; review only unusual offsets
Full re-validation	Quarterly, whole twin	Quarterly on a **risk-stratified 10 per cent sample**
Physical changes	6 a year, a form	Same form, now attached to the bench-booking system
Purpose question	Ask the researcher	Ask **five researchers a quarter**, across groups

The operating bill grows sublinearly: Chapter 14's per-twin lines are paid three times rather than twenty, and the per-asset lines are the ones Sec. 15.3 converted.

15.5.4 What the facility gains that the bench could not

One thing, and it is worth stating because it is the only genuine upside of scale in this example: **180 pots is a population, and a population makes the drought group's experiment interpretable.** Chapter 7's residual spread of 2.0 was measured on one bench in one season; across 180 pots and four groups it can be measured as a distribution, and an unusual pot becomes detectable against its peers rather than against a threshold somebody chose.

That is Sec. 15.3.5's technique used for science rather than for cost, and it is the reason population methods survive contact with people who do not care about engineer-days.

15.6 Faded example: the wind farm inside an energy system

Chapters 4 through 14 followed one turbine and then eighty. Now the farm is one participant in a regional energy system that also contains a grid operator's twin, a weather service, and two other farms. Two parts are worked; four are yours.

(a) This is the mesh case, and it is the first time in the book that it is – worked. Sec. 15.4.1 said the mesh needs a reason to exist, and here it is: **the participants belong to different organisations with no natural centre.** The grid operator is not the farm's owner, the weather service sells to all of them, and the farms are competitors who must nevertheless coordinate curtailment.

Count: five participants, so `5 x 4 / 2 = 10` bilateral agreements against 5 standard adoptions. At Chapter 13's figures that is 40 days against 50 – **bilateral is still cheaper at five**, and Sec. 15.4.1's break-even at six is doing real work rather than decorating a page. **The moment a sixth participant joins, the answer flips**, and the honest planning position is to adopt before that rather than after.

(b) The composite's freshness is bounded by the weather service – worked, because it is the counter-intuitive one. Sec. 15.4.3's rule 2 says a composite is as fresh as its stalest input. The farm's own twin runs at Chapter 9's three rates, the fastest of them 2 kHz. The weather input arrives every ten minutes at best and is a *forecast*, so its effective age is negative and its uncertainty dominates.

So a composite curtailment decision cannot be more current than ten minutes, no matter what the turbine twins do – and any design that implies otherwise has mistaken the freshness of a component for the freshness of the whole. Chapter 2's age-of-twin, defined for one stream, is here a property of a composition.

Now yours.

(c) Apply Sec. 15.4.3's three rules to the composite's credibility argument. Which component's expiry would invalidate a curtailment decision most quietly, and what column does Chapter 14's expiry register need?

(d) Sec. 15.4.2 warned that composite uncertainty is usually computed assuming independence. Name three ways the eighty turbines' prediction errors are correlated, and say what each does to the farm-level uncertainty. *Hint: Chapter 7 Sec. 7.6.1's four sources, read across assets rather than across time.*

(e) The grid operator asks the farm to publish a trust vector [7]. Using Chapter 7's five-part argument, say what you would publish, what you would refuse to publish, and how a counterparty could check the parts you do publish.

(f) Sec. 15.4.4's data-space question, applied. Is there data here that cannot move, and a party who needs what it would tell them? Answer for each of the five participants, and say whether the answer justifies the arrangement.

15.7 Posed problem: the fleet proposal

No solution is given. It is the last posed problem in the book and it asks for everything.

The situation. A rail operator runs 1,200 passenger vehicles across four depots. Each vehicle has door systems, Heating, Ventilation and Air Conditioning (HVAC), traction and bogies, all instrumented to some degree. Today there is one twin: a successful proof-of-concept covering **one vehicle**, built by a small team, detecting door faults, with a credibility argument and a nine-month payback that the board liked.

The board has asked for "the same thing, for the fleet", and has approved a budget on the assumption that it costs 1,200 times the proof-of-concept.

Produce a proposal of no more than six pages containing:

1. **How many twins**, per Sec. 15.1.4, with the three questions answered explicitly. Note that four subsystems and 1,200 vehicles do not necessarily give either 4 or 1,200 or 4,800.
2. **The unit conversion** (Sec. 15.1.3): what the proof-of-concept's cost actually contains, split into per-twin and per-asset, before any multiplication.
3. **The commissioning bill** (Sec. 15.2.5) under the board's assumption and under a population approach, with the factor stated. **This is the number the board's assumption gets wrong**, and it is the proposal's central point.
4. **The scaling table** (Sec. 15.2): every cost line sorted into fixed, linear and worse-than-linear, with the quantity that binds first identified and its arithmetic shown.
5. **The population practices** you would adopt (Sec. 15.3), and **for each, what it trades away**, stated as plainly as Sec. 15.3.2 to Sec. 15.3.6 state theirs.
6. **The alert arithmetic** (Sec. 15.2.4) for 1,200 vehicles, against the staffing actually available, with your ranking design and its blind spot.
7. **Whether this is an ecosystem** (Sec. 15.4): are there counterparties – a maintenance contractor, a regulator, a leasing company, the manufacturer – and is it hub or mesh?
8. **One paragraph telling the board what they got wrong**, in their units rather than yours, and what the corrected payback looks like.

What a good answer looks like. It does the unit conversion before multiplying anything, because that is where the board's error is. It reaches a commissioning figure that is large and defensible, and it offers the population alternative rather than only the objection. It says which quantity binds first and shows the arithmetic. It resists making the twin count equal the asset count. And its last paragraph is honest in Chapter 1's terms: **a proof-of-concept's payback does not survive multiplication, because the numerator scales and a substantial part of the denominator was paid once** – which is not a reason to refuse the programme, but is the reason to size it properly before the board hears a number twice.

15.8 Summary

Seven things, tied to the five objectives.

1. **How many twins is a design decision, and it comes first** (Sec. 15.1). Chapter 14's operating bill is per *twin*, and the demonstrator's twin covers twelve pots – so "scale to 400 assets" is not "multiply by 400" until somebody has decided there are 400 twins. Roughly 11.5 of Chapter 14's 17.3 days are per-twin and about 4 per-asset, which means **the cheapest way to cover more assets is usually to put them inside one twin.** The limit is the decision, not the hardware: one twin cannot have two intended-use statements. *(Objective 1.)*
2. **Three columns, and the third is short** (Sec. 15.2). Fixed: the build, upgrades, the register's structure, a standard's `F`. Linear: ingest, alerts, sensor events, context rows. Worse: human attention, and integrations in the mesh case. *(Objective 2.)*
3. **Human attention is the quantity that binds first** (Sec. 15.2.4). The district-heating twin's 2,080 alerts a year is 208 engineer-days of triage demand against about 22 available – **nine to one**, which explains a fact Chapter 11 reported and could not explain: the operators "mostly ignore" the alerts because ignoring eight

in nine is the only available response. *(Objective 2.)*

4. **Commissioning, not operating, is what actually breaks** (Sec. 15.2.5), and no earlier chapter costed it. Chapter 7's per-asset work at 3 engineer-days times 400 assets is **1,200 days, about 5.5 engineer-years before anything runs** – two orders of magnitude above Chapter 14's annual operating bill. Population methods take it to about 120. *(Objective 2, 3.)*
5. **Five practices convert per-asset work into per-population work, and each has a stated price** (Sec. 15.3): sampled validation is blind to rare conditions on unsampled assets; ranking is blind to the small persistent fault; kinds-and-instances requires the relationship layer; population calibration models an atypical asset *worse* while making it look normal; sampled purpose questions find widespread drift and miss localised drift. **Naming the price is the point** – Chapter 11 Sec. 11.4.7's confirmation-defeats-spread-not-bias, generalised. *(Objective 3.)*
6. **Chapter 13's rule, in the mesh case, flips** (Sec. 15.4.1). `N(N-1)/2` bilateral agreements against `N` adoptions gives break-even at **six mutually-interoperating twins**, and eighty-to-one at four hundred. Same rule as Chapter 13's, opposite answer, and the variable is still how many people you must agree with. Most fleets are hubs and this does not apply to them. *(Objective 4.)*
7. **A composite's credibility argument cites rather than restates, is bounded by its weakest component on every axis separately, and must have component expiries wired to it** (Sec. 15.4.3) – the last of which is the most likely defect in any composed twin. And the honest close: composing uncertainty correctly, credibility across organisations, and integrating separately-built models are **open problems**, and five of the six named obstacles to composite twins are not engineering problems at all. *(Objectives 4, 5.)*

15.9 The book, closed

15.9.1 What you can do now

Not build a twin on your own. **Hold up your end of building one**, which was the goal stated on the first page and is a different and more useful thing.

Concretely, and each of these is a chapter: name the decision a twin would serve and the metric it would move; classify what somebody is proposing and say whether it is a twin at all; draw the seven components and say which the project needs; ask a modelling expert five questions that change their answer; say what a simulation run costs before anybody builds it; name what will have to solve each part of a proposed twin and predict where the compute goes; demand a credibility argument and know what a good one contains; place machine learning where it belongs and refuse it where it does not; build a connector that does not lie to you; design a store that can answer what the twin believed last August; ship services with three outcomes; price a platform; count counterparties before adopting a standard; write an expiry register and cost the operating year; and decide how many twins a fleet is.

The through-line, if you keep one sentence from fifteen chapters:

> **Derive it from the decision.** Fidelity, sampling interval, accuracy target, service set, platform, standard, twin boundary – every one of them was settled

in this book by asking what decision the twin serves and what being wrong costs, and never by asking what was technically possible.

15.9.2 What this book did not teach

Stated plainly, because a reader who overestimates what they now know is worse off than before.

You cannot build a model. Part II was explicit and repeated: it teaches what each kind of model is *for*, what it costs, and what to ask. You cannot derive a Runge-Kutta step, mesh a finite-element problem, implement a particle filter, or write a Modelica library.

You cannot do the statistics. No Bayesian calibration, no confidence intervals, no sensitivity indices. Chapter 7 gave you an average, a spread, and a hold-out, and said so.

You cannot write a safety case. Chapter 7 Sec. 7.7.6 named it as a different artifact with a different owner and stopped there. If your twin closes a loop on something that can hurt someone, you need that person.

You have not been told which product to buy. Chapters 12 and 13 refused, with reasons.

And the numbers in this book are illustrative. The demonstrator's sensors, calibration values, alert rates and costs were constructed to be internally consistent and to teach a method. Chapters 4, 7, 8, 9, 11 and 12 each said so at the point of use. The method transfers; the values do not.

15.9.3 The demonstrator, honestly

The book's running example is a real physical twin: a plant-pot rig with sensors, pumps and a documented HTTP API, which is why Chapter 1 could describe its hardware from its own source rather than from an idealisation.

And the digital twin of it does not exist. Chapter 13 Sec. 13.5.3 said this in passing and it deserves saying properly at the end: the exemplar this book has used for fifteen chapters demonstrates **what a physical twin looks like before anyone twins it** – which is the state most readers will actually meet, and the state that published exemplars usually skip.

So the book ends with its own worked example unbuilt, on purpose. Every chapter's final exercise has been a step toward building it, and exercise 15.10.10 is the last one.

15.9.4 Where to go

Read somebody else's twin. Chapter 13 Sec. 13.5.3 named three that are published, documented and runnable: the incubator, with a tutorial [11], a calibration case study [12] and a survey that builds the *same* case study across five frameworks [13]; and GreenhouseDT, built around a knowledge graph that records structure and its changes [2]. An afternoon with one of these is worth more than another book.

Then build the smallest thing that serves a decision. Chapter 1's method and Chapter 2's honest sentence – *this is a digital shadow, it will tell you within eight hours*

that pot 7 stopped receiving water, it will not water pot 7 – are the whole of a good first project.

And expect the second year to be the interesting one. Chapter 14 is the chapter most readers will need soonest and appreciate last.

In the literature, for this chapter and for what comes after it.

- *Scale as a research problem:* [14] on moving twins from the artisanal to the industrial, which is the reference Chapter 1 parked for here and which names validation speed as the binding constraint; [15] on the open agenda, including at fleet scale.
- *Composition, and the vocabulary around it:* [5] on composite twins and the six obstacles – trust, interoperability, governance, ownership, security, privacy; [1] on hierarchical composition as an architectural capability; [16] on composability as the route to enterprise scale, from Chapter 12; [17] and [18] on composing twins across the edge-to-cloud continuum, which Chapter 2 parked for here.
- *What is unsolved:* [6] is the one to read – nine integration challenges with a call to the community, and the reference Chapter 4 parked for this chapter.
- *Populations:* [3] on population-based structural health monitoring and transfer across a fleet; [4] on a calibration and updating process that scales to an entire fleet; [19] on combining sources of information across wind farms.
- *Ecosystems and exchange:* [7] on machine-readable trust between twins, which Chapter 7 pointed here; [10], [9] and [8] on cross-organisation exchange; [20] on the platform view, which Chapter 3 said would come back here.
- *Consulted, not drawn on above:* [21], [22] and [23] on system-level and interoperability framings, and [24] on distributed twin infrastructure.

15.10 Exercises

Solutions or hints follow. Each exercise names the objective it exercises.

15.10.1 *(Objective 1.)* For each, say how many twins and why: (a) 30 identical air handling units in one building, all monitored for filter blockage; (b) the same 30 units, plus a question about total building energy use; (c) 30 units across six buildings owned by five different landlords; (d) two units that are identical hardware, one in a data centre and one in a concert hall.

15.10.2 *(Objective 1.)* Chapter 14's bill was 17.3 engineer-days for a twin covering twelve pots. Your colleague proposes twelve twins, one per pot, "for cleanliness". Compute both, then give the one argument *for* their proposal that is not about cost.

15.10.3 *(Objective 2.)* Sort into fixed, linear or worse: (a) the credibility argument's page count; (b) the number of expiry-register rows; (c) the time spent reviewing the register monthly; (d) the number of bilateral data agreements in a hub; (e) the same in a mesh; (f) the number of people who must remember to report a physical change.

15.10.4 *(Objective 2.)* A fleet of 250 assets produces 6 human-facing alerts per asset per year. Triage is 0.15 engineer-days each. Available attention is half of one person. Compute demand, supply and the ratio, then state what must change and by how much for the service to be usable.

15.10.5 *(Objective 2.)* Estimate the commissioning bill for 250 assets at Chapter 7's per-asset cost, then for a population approach, and state the one assumption the population figure depends on and how you would test it before committing.

15.10.6 *(Objective 3.)* For each population practice, name what it trades away and give one situation where the trade is unacceptable: (a) sampled validation; (b) ranking instead of notifying; (c) population calibration; (d) sampled purpose questions.

15.10.7 *(Objective 3.)* Sec. 15.3.5's step 3 flags assets with unusual offsets. Design the check: what counts as unusual, what a reviewer looks at, and what the two possible conclusions are. Then say what happens if 40 per cent of assets are flagged.

15.10.8 *(Objective 4.)* Nine organisations, each with a twin, must all exchange with each other. With `I` = 6 and `F` = 12 engineer-days, compute both options. Then compute the break-even number of participants, and say what would make you recommend bilateral agreements anyway.

15.10.9 *(Objective 4 and 5.)* A composite twin's claim is "the network will meet demand for the next six hours". Its components are 400 substation twins (as of 10 minutes ago), a demand forecast (issued hourly), and a weather feed (updated every 30 minutes). State the composite's freshness, its envelope, and its expiry, per Sec. 15.4.3's rule 2 – and name one thing about its uncertainty you cannot state at all.

15.10.10 *(Objectives 1-5, and the last exercise in the book.)* Build it. Take the plant-controller rig, or any physical thing you have access to that produces data and has a decision attached. Work Chapter 1's method to a value case, Chapter 3's anatomy to a component list, and Chapters 9 to 11 to a connector, a store and one service. Then write Chapter 7's credibility argument and Chapter 14's expiry register for what you built. **If the value case fails, stop – and that is a successful outcome**, because Chapter 1 Sec. 1.7 said so and it cost you an afternoon instead of six months.

Solutions and hints

15.10.1. (a) One twin, 30 bindings – same decision, near-identical assets. (b) **Two**, or one twin plus a composite: the building-energy question is a different question about a different thing, per Sec. 15.1.4's third test. (c) **At least two considerations collide**: the decision is the same, so technically one twin would serve – but five landlords means five counterparties, five sets of access rules and possibly five credibility arguments, so the *organisational* boundary forces the split even though the engineering does not. This is the exercise's point. (d) **Two twins.** Identical hardware, entirely different operating envelopes and therefore different models, different thresholds and different consequences of failure. Sec. 15.1.3's limit is about the decisions and the conditions, not the parts.

15.10.2. One twin: 17.3 days. Twelve twins: the per-twin lines (about 11 days) are paid twelve times, so roughly `12 x 11 + 6 = 138` days – about eight times more. **The argument for it that is not about cost:** twelve twins can be retired, re-scoped or re-validated independently. If one pot joins a different experiment (Sec. 15.5.1's drought group), a per-pot twin follows it and a shared twin has to be split. **Optionality has a price and here it is eight times the operating cost**, which is a fine thing to decline explicitly.

15.10.3. (a) Fixed – Chapter 7's two pages describe a claim, and a fleet claim is still two

pages. (b) Linear – more assets, more rows. (c) **Fixed**, and this is the useful one: the monthly review reads *categories* of row, not rows. Design it that way and it stays fixed. (d) Linear. (e) Worse – `N(N-1)/2`. (f) Linear in people, and Sec. 15.2.4's honourable mention: the detection mechanism does not improve with `N`, so the failure rate per event is roughly constant and the absolute number of missed changes grows.

15.10.4. Demand: `250 x 6 x 0.15 = 225` engineer-days. Supply: half a person is about 110 days, though realistically far less is spent on triage – take 30 days as the honest figure. **Ratio about 7 to 1.** What must change: the alert count, by roughly a factor of seven, and Sec. 15.3.3 says the only lever that reduces *volume* rather than improving the ratio is to stop notifying per asset. A top-ten daily list is about 2,500 items a year seen, of which the operator works whatever they can – **note that this does not reduce the alerts, it reduces the ones a human is asked to look at**, and the difference matters for the credibility argument.

15.10.5. `250 x 3 = 750` engineer-days, about 3.4 engineer-years. Population: `20 + 250 x 0.25 = 82.5` days, plus review of the unusual, so about 95 – a factor of about eight. **The assumption:** that the 250 assets form a population, meaning a model fitted across them with per-asset offsets predicts each about as well as a per-asset fit would. **How to test it before committing:** commission ten assets the expensive way, fit a population model on those ten, and compare the population-plus-offset predictions against the individually-fitted ones on held-out windows. That costs 30 days and settles a 650-day question.

15.10.6. (a) Trades away detection of rare-condition failures on unsampled assets; unacceptable where a rare condition is the dangerous one – a flood-defence asset validated only in dry weather. (b) Trades away the small persistent fault; unacceptable where slow degradation is the failure mode you are paid to catch, which is most predictive maintenance. (c) Trades away accuracy on atypical assets *while making them look normal*; unacceptable where the atypical asset is the important one – the oldest unit, the one in the harshest location. (d) Trades away localised drift; unacceptable where one user's misuse is high-consequence, such as a single clinician using a population tool for an individual decision.

15.10.7. Unusual: the offset falls outside the bulk of the population's offsets – Chapter 7's *spread*, applied across assets rather than across time, with a stated multiple. What the reviewer looks at: the asset's raw data and its context row, in that order. **The two conclusions are the whole point**: either the asset is genuinely different, in which case it needs its own parameters and possibly its own model, or the data is wrong, in which case it is a Chapter 9 problem. If 40 per cent are flagged, **the population assumption has failed** – these are not one population, and the right response is to look for the natural sub-populations in the context model (vintage, location, supplier) rather than to loosen the threshold.

15.10.8. Bilateral: `9 x 8 / 2 = 36` agreements, `36 x 6 = 216` engineer-days. Standard: `9 x 12 = 108` days. **Standard wins by two to one.** Break-even: `N(N-1)/2 x 6 = N x 12`, so `(N-1) x 3 = 12`, `N = 5`. What would make you recommend bilateral anyway: **if only a few of the 36 pairs actually need to exchange anything.** The mesh arithmetic assumes everyone talks to everyone, and Sec. 15.4.1 warned that most fleets are hubs. Count the pairs that genuinely exchange before believing 36.

15.10.9. Freshness: **bounded by the hourly demand forecast**, so the composite is at best an hour old regardless of the substations' ten minutes – and Sec. 15.6(b) is the same finding on a different system. Envelope: the intersection of the three components' envelopes, which is narrower than any of them and is a different set of conditions from each. Expiry: the earliest of the three, and it is probably the weather feed's. **What you cannot state about its uncertainty:** how correlated the 400 substation errors are. They share weather, share a calibration procedure and share suppliers, so they are certainly not independent – and without knowing the correlation you cannot combine 400 individual ranges into a network range at all. Sec. 15.4.2 named this and Sec. 15.4.5 lists it as unsolved.

15.10.10. No solution, and no hint. This is the book's last exercise and it is the only one whose output is a system rather than a document. **The most likely outcome is that you stop at Chapter 1's value case, and that is the outcome this book most wants for you** – because an afternoon that prevents six months is the highest-return thing in these fifteen chapters, and it is available to you now in a way it was not before you started.

15.11 References

[1] M. Martinelli, J. Zhang, A.-K. Splettstoßer, M. Picone, M. Lippi, and A. Wortmann, "Hierarchical Digital Twin Ecosystem for Industrial Manufacturing Scenarios," in *2024 50th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, Aug. 2024, pp. 56–63. doi: 10.1109/SEAA64295.2024.00018.

[2] E. Kamburjan *et al.*, "GreenhouseDT: An Exemplar for Digital Twins," in *Proceedings of the 19th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, in SEAMS '24. New York, NY, USA: Association for Computing Machinery, Jun. 2024, pp. 175–181. doi: 10.1145/3643915.3644108.

[3] P. Gardner, L. A. Bull, J. Gosliga, N. Dervilis, and K. Worden, "Foundations of population-based SHM, Part III: Heterogeneous populations – Mapping and transfer," *Mechanical Systems and Signal Processing*, vol. 149, p. 107142, Feb. 2021, doi: 10.1016/j.ymssp.2020.107142.

[4] M. G. Kapteyn, J. V. R. Pretorius, and K. E. Willcox, "A probabilistic graphical model foundation for enabling predictive digital twins at scale," *Nature Computational Science*, vol. 1, no. 5, pp. 337–347, May 2021, doi: 10.1038/s43588-021-00069-0.

[5] P. Kuruppuarachchi, S. Rea, and A. McGibney, "An architecture for composite digital twin enabling collaborative digital ecosystems," in *2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, IEEE, 2022, pp. 980–985. Accessed: Feb. 25, 2025. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9776073/

[6] B. Combemale *et al.*, "On the Challenges of Integrating Digital Twins," in *2nd International Conference on Engineering Digital Twins (EDTconf 2025)*, 2025. Accessed: Sep. 29, 2025. [Online]. Available: https://inria.hal.science/hal-05221809/

[7] "Assuring Trustworthiness in Dynamic Systems Using Digital Twins and Trust Vectors." Digital Twin Consortium, May 2024. Available: https://www.digitaltwinconsortium.org/assuring-trustworthiness-in-dynamic-systems-using-digital-twins-and-trust-vectors-form/

[8] J. M. Bernabé Murcia, E. Cánovas, J. García-Rodríguez, A. M. Zarca, and A. Skarmeta, "Decentralised Identity Management solution for zero-trust multi-domain Computing Continuum frameworks," *Future Generation Computer Systems*, vol. 162, p. 107479, Jan. 2025, doi: 10.1016/j.future.2024.08.003.

[9] K. Gleich, S. Behrendt, M. Hörger, M. Benfer, and G. Lanza, "An Asset Administration Shell-Based Digital Product Passport as a Gaia-X Service," *Procedia CIRP*, vol. 127, pp. 224–229, Jan. 2024, doi: 10.1016/j.procir.2024.07.039.

[10] P. Singh,..N., M. Beliatis, and M. Presser, "Data-Driven IoT Ecosystem for Cross Business Growth: An Inspiration Future Internet Model with Dataspace at the Edge," *INTERNET 2024 :*, Mar. 2024.

[11] C. Gomes *et al.*, "Digital Twin Tutorial: The Incubator Case Study," in *Engineering Trustworthy Software Systems: 6th International School, SETSS 2024, Chongqing, China, April 14–21, 2024, Tutorial Lectures*, J. P. Bowen, C. Gomes, and Z. Liu, Eds., Singapore: Springer Nature, 2025, pp. 68–101. doi: 10.1007/978-981-96-4656-2_3.

[12] B. J. Oakes *et al.*, "Case Studies in Digital Twins," in *The Engineering of Digital Twins*, J. Fitzgerald, C. Gomes, and P. G. Larsen, Eds., Cham: Springer International Publishing, 2024, pp. 257–310. doi: 10.1007/978-3-031-66719-0_12.

[13] S. Gil, P. H. Mikkelsen, C. Gomes, and P. G. Larsen, "Survey on open-source digital twin frameworks–A case study approach," *Software: Practice and Experience*, vol. 54, no. 6, pp. 929–960, Jun. 2024, doi: 10.1002/spe.3305.

[14] S. A. Niederer, M. S. Sacks, M. Girolami, and K. Willcox, "Scaling digital twins from the artisanal to the industrial," *Nature Computational Science*, vol. 1, no. 5, pp. 313–320, May 2021, doi: 10.1038/s43588-021-00072-5.

[15] *Foundational Research Gaps and Future Directions for Digital Twins*. Washington, D.C.: National Academies Press, 2024. doi: 10.17226/26894.

[16] P. van Schalkwyk and D. Isaacs, "Achieving Scale Through Composable and Lean Digital Twins," in *The Digital Twin*, N. Crespi, A. T. Drobot, and R. Minerva, Eds., Cham: Springer International Publishing, 2023, pp. 153–180. doi: 10.1007/978-3-031-21343-4_6.

[17] A. Barbone, S. Burattini, M. Martinelli, M. Picone, A. Ricci, and A. Virdis, "Digital Twin Continuum: A Key Enabler for Pervasive Cyber-Physical Environments," in *2024 33rd International Conference on Computer Communications and Networks (ICCCN)*, Jul. 2024, pp. 1–9. doi: 10.1109/ICCCN61486.2024.10637565.

[18] M. S. Gill, J. Zhang, A. Wortmann, and A. Fay, "Toward Automating the Composition of Digital Twins Within System-of-Systems," in *2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2024, pp. 1–4. doi: 10.1109/ETFA61755.2024.10710740.

[19] L. A. Bull *et al.*, "Data-Centric Monitoring of Wind Farms: Combining Sources of Information," in *Data Driven Methods for Civil Structural Health Monitoring and Resilience*, CRC Press, 2023.

[20] P. Talasila, P. H. Mikkelsen, S. Gil, and P. G. Larsen, "Realising Digital Twins," in *The Engineering of Digital Twins*, J. Fitzgerald, C. Gomes, and P. G. Larsen, Eds., Cham: Springer International Publishing, 2024, pp. 225–256. doi: 10.1007/978-3-031-66719-0_11.

[21] H. Marah and M. Challenger, "(Re-)Engineering Digital Twins Towards Federation: Vision and Roadmap," in *Leveraging Applications of Formal Methods, Verification and Validation. Software Engineering Methodologies*, T. Margaria and B. Steffen, Eds., Cham: Springer Nature Switzerland, 2025, pp. 60–81. doi: 10.1007/978-3-031-75387-9_5.

[22] A. Budiardjo and D. Migliori, "Digital twin system interoperability framework," Tech. rep. Digital Twin Consortium, East Lansing, Michigan, 2021. Accessed: Feb. 25, 2025. [Online]. Available: https://www.digitaltwinconsortium.org/pdf/Digital-Twin-System-Interoperability-Framework-12072021.pdf

[23] E. Altamiranda, "A System of Systems Foundation for Digital Asset Lifecycle Management," in *Digital Twin: Fundamentals and Applications*, S. Sabri, K. Alexandridis, and N. Lee, Eds., Cham: Springer Nature Switzerland, 2024, pp. 59–87. doi: 10.1007/978-3-031-67778-6_4.

[24] D. McKee and D. Dokter, "DISCS: An Approach for Accelerating the Development of Digital Twins for Smart Cities," in *Digital Twin: Fundamentals and Applications*, S. Sabri, K. Alexandridis, and N. Lee, Eds., Cham: Springer Nature Switzerland, 2024, pp. 31–58. doi: 10.1007/978-3-031-67778-6_3.