Capturing and Reasoning with Quality Attenuation
the ΔQ Ecosystem

Neil Davies

February 14, 2024

Contents

1 Quality Attenuation and outcome

- Outcome: something that is a day-to-day / commercial interest in what occurs in a system, at most once, over a particular interval of time. An outcome being in progress is knowable.
- The interest (after some notion of correctness) in the *outcome* is in its timeliness - how

long it is in progress, or how quickly it finishes given it has started. It is not property of outcomes that they must eventually complete.

- There may be a population of outcomes of the same sort. It is not unreasonable to discuss the distribution of the timeliness of such a population, or its frequency.
- Delivering an outcome incurs costs (even if it doesn't complete), cost is not necessarily a simple scalar.

A 'perfect' outcome is defined to be an outcome that occurs with zero duration.

1.1 Outcomes in the context of translocation

Any outcome that is extended in space is, necessarily, extended in time and thus is less than perfect, we can say that its quality is attenuated. Only outcomes that occur at a single point in space can be 'perfect'. Any information translocation is therefore, necessarily, attenuated.

1.2 Outcomes in the context of computation

... similar words to above but about computation ...

2 Observables

Only certain aspects of the system under consideration are observable. In particular the beginnings and ends of outcomes are observable.

Perfect knowledge of past light cone of an outcome[1], yields certainty of quality attenuation. Imperfection in this knowledge yields uncertainty.

This is not the only type of uncertainty that can exist, even in the presence of perfect contemporaneous knowledge (i.e. complete knowledge of the light cone associated with the leading edge of the outcome), there will be future events (which sit in the difference between the light cones of the trailing edge and the leading edge of the outcome).

There is a familily of morphisms between notion of quality attenuation, observables, population of observables, and ΔQ.

3 Relationship between concepts

ΔQ ΔQ has its foundations in the notion of an improper random variable — something that captures a continuous quantity (such as delay or some other scalar) combined with the discrete concept of non-occurrence (such as loss, discard, failure or divergence). A typical way of thinking about a ΔQ value is: *Given event A occurred what is the probability of event B ever occurring and what is the distribution of delay between those two events?*.

ΔQ Algebra The collection of distinguished values and operators that form an algebra that permits the construction and manipulation of ΔQ values. Namely:

[1] Noting that the past light cone of an outcome relates to the trailing edge of that outcome.

- $\varnothing$ perfection – no delay / unconditional success – the *unit* for this algebra.
- $\perp$ bottom – unbounded delay / unconditional failure – the *absorbing element* for this algebra.
- $\rightleftharpoons$ probabilistic choice. ($left \overset{p}{\underset{q}{\rightleftharpoons}} right$) represents a Bernoulli Choice which chooses between *left* and *right* in the ratio of $p : q$, that is the *left* choice with probability $p/(p+q)$.
- $\oplus$ convolution. Convolution captures the notion of sequential composition.
- $\mathbb{D}$ The model for the underlying continuous quantity (a proper random variable). The delay model.

ΔQ Calculus ΔQ Algebra + behaviour. This combines the ΔQ Algebra with a suitable representation of the notion of behaviour, e.g Process Algebra or Labelled Transition System. This calculus underpins refinement as it supports both abstraction (considering some emergent outcome from a component's implementation) and reification (decomposing the delivery of an outcome into one or more collaborative components).

The delivery of an outcome is (see above) is, in itself, a ΔQ. This gives a relationship between behaviour, where the behavioural steps have associated ΔQ, and the ΔQ of the outcome(s) of that behaviour.

Thus the ΔQ calculus can be used to form statements which can capture concepts relating to feasibility & risk:

- What is the probability that an outcome, $\mathbb{O}$, will occur within a time, t?
- What is the probability distribution of the time to complete an outcome, $\mathbb{O}$? What is the likelihood that $\mathbb{O}$ will complete within $t_0, t_1, t_2, \ldots$. What is the likelihood that it will ever complete (i.e as $t \to \infty$)?
- For given outcome,$\mathbb{O}$, that has a ΔQ, what is its average time to complete?, its median? the variance in completion time? it probability of 'failure' (not completing)? its 95-centile of completion (if it exists)?

Such concepts can be flowed through abstraction and reification steps, and can be applied at multiple stages of a system's life-cycle, e.g. it can be used to capture design choice interactions with performance risks, analyse implemented components (to inform integration risks and/or in-life optimisation decisions) as well as providing a quantitative framework for monitoring overall system performance during its deployment (supporting proactive management and capacity planning).

QTA Quantitative Timeliness Agreement. The operation of any system requires some infrastructure, the *supply*. As a process executes to deliver an outcome, $\mathbb{O}$, it will place a *demand* on that infrastructure. A QTA captures the demand and supply aspects that underpins $\mathbb{O}$ occurring with a given ΔQ.

From the supply perspective, the QTA can be seen as making a refinement of of the notion of ΔQ in the algebra and calculus. In a QTA context, the ΔQ is referring to a delivered (or deliverable) performance outcome from some implementation of *supply*.

This refinement from the abstract notion of outcome, in turn, has implications on how any given supply is designed, constructed and operated.

From the demand perspective, the QTA is capturing a measure (again stochastic in nature) of how much load is being based on infrastructure. The load could be on one (or more) resources that are contained with the infrastructure. For example in distributed computing that may contain the load placed on the communications network as well as the computational load on the processing elements within that network.

Thus a QTA is capturing the relationship between *supply*, *demand* and ΔQ. For example, consider the time to download a web page of a given size, the quicker that outcome is completed the more intense the demand. If a a given supply (the data transport path between the source and destination) has a increased delay then time taken to complete the outcome will be increased which, in turn, reduces the intensity of the demand. If a particular supply has an increased loss rate that will both increase the time to complete the outcome (which will decrease the intensity of demand) but also create more demand (as information will need to be retransmitted to mitigate the loss). Although this is a qualitative description, the QTA captures the relationships quantitatively, thus allowing for formulation of questions such as:

- Given a ΔQ measure of the outcome, if the delivered ΔQ of the supply changes by x how much does the ΔQ of the outcome change?

  This is technical aspect of the issue of how supply side changes influence the delivered UX/QoE (User Experience/Quality of Experience).

- Given the desire to deliver a particular outcome (either to improve its delivered performance, or before entering into some new commercial arrangement) with a specific ΔQ, what is the ΔQ required from the supply and how much demand will it place on that supply?

  This is the technical aspect of how to create agreements between parties in digital supply chains. The supply-side ΔQ capturing the delivery requirement from the infrastructure, where as the demand-side measure captures the offered-load on that infrastructure.

In capturing this relationship the QTA is also creating specific points of measurement, their range of likely values and how that relates to a measurable outcome.

QTA and resources The notion of a resource in a QTA is very general and the particular resources of interest will vary according to the particular domain being modelled.

In the general distributed computing case there are several resource types that come to mind: CPU cycles, network data transport, memory footprint and disk capacity as exemplars. In considering how QTAs aggregate one distinguishing aspect is what is the response of the infrastructure when a QTA makes demand for that resource and that resource is not idle. This is best illustrated with some examples:

- A demand for memory capacity. If there is no allocatable memory then the infrastructure will respond with $\perp$. Memory is a *threshold* constrained resource. While below the threshold the supply's ΔQ is $\varnothing$, once above the threshold the ΔQ is $\perp$.

- A demand for certain quantity of CPU time. This is a demand for a share of an (ephemeral) resource. Whether the infrastructure can deliver the supply with the desired ΔQ is a question of if the demand can be satisfied within the appropriate time-scale. CPU time is a *schedulability* constrained resource. If the demand can not be met within the time-scale, the infrastructures (typical) response would be to deliver the supply with a increased ΔQ.

It is reasonably clear that disk capacity is threshold resource. However which category network data transport falls into depends on how that resource has been constructed. Where that data transport is being discretely allocated, as in the case of light-paths or TDM capacity, then the transport resource falls into the threshold resource category. Where the data transport is statistically multiplexed resource (such as broadband) it typically falls into the schedulability category.

It is uncommon for a system to explicitly engage with the notion that infrastructure resources have different styles of demand response. Often such responses only manifest themselves late in the project development or post deployment. Consequentially, the system's reaction to such responses can lead to severe project or service execution issues.

Although infrastructure's capacity can be increased, there is a limit to how dynamic that can be. This has consequences for short term handling (i.e. how does the system gracefully degrade) and capacity planning (i.e. how and when to provision increased infrastructure).

Although these seem a very diverse set of issues, because they all manifest themselves as change in delivered ΔQ and that, in turn, can be translated to (increasing higher level) ΔQ effects on outcomes, the concept of opportunity-cost can be used to unify them for presentation to stakeholders.

4 Quality Attenuation - ΔQ

Quality attenuation / impairment as a privation, like silence and darkness.

- Relationship with passage times in process algebras, primacy *observation* in the ontology, notion of *bisimulation.*
- Notion of 'improperness' - probability mass and (in this case) its non-conservation
- ΔQ as a 'conserved' quantity - how this concept captures the quality attenuation / impairment.
- How ΔQ is the consequence of resource sharing - how it captures the effects of queueing systems
- How ΔQ 'accrues' and how that accrual captures the notion of both 'journey' and 'evolution'.
- Use of CDF as measure of *better*, notions of *slack* and *hazard.*
- Models of the delay/impairment component - use of Uniform Distribution (comparison with neg-exponential phase space)

5 ΔQ Algebra properties

5.1 Probability Mass

5.1.1 Intangible Probability Mass

5.1.2 Tangible Randomness

5.2 Properties of $\varnothing$ and $\perp$

5.3 Properties of $\leftrightharpoons$

5.4 Properties of $\oplus$

5.5 Delay Model

- Uniform Delay as basis function for finite times
- Representation of uniform delay as Dirac delta (δ) and Uniform from zero ($\sqcap$)
- Delay model properties under $\leftrightharpoons$ and $\oplus$

5.6 ΔQ basis set

The role of $\Delta Q_{|G}$, $\Delta Q_{|S}$ and $\Delta Q_{|V}$ for capturing different contributors of quality attenuation in real world scenarios (e.g. data networking, vector processing or any service facility that has a set up time and a rate of service once set up).

6 ΔQ Calculus

7 QTA

- QTA as a performance design/analysis building block
- QTA aggregation and disaggregation - the application Erlang.
  - Time scale effects - the curse of ‘bandwidth‘
  - heterogeneous and homogeneous aggregation/disaggregation; positive/negative and random correlation.
  - Mapping of aggregated QTA requirements into treatment classes — robust simplification of heterogeneous demands into low-complexity homogeneous delivery.
- QTA and the digital supply chain.
- QTA and overbooking hazards.
- QTA and graceful degradation.

8 QTA and resources

- Graceful degradation
- Capacity planning as probability of failing to meet a QTA

- Scheduling mechanisms for ephemeral resources — how that interacts with the resource's threshold/schedulability allocation constraints.
- Other resources and their allocation
- Capturing the opportunity cost of supporting a QTA on a particular resource.

