```
val tickets = UUIDPool() with ErrorTolerance(0.05)

def viewEvent(event: UUID) = {
  val range: Interval[Int] = tickets(event).remaining()
  if (range.min > 1000)
    display("Many tickets left.")
  if (range contains 0)
    display("Warning: tickets may be sold out.")
  //...
}

def purchase(event: UUID) = {
  tickets(event).take() match {
    case Consistent(Some(t)) | Inconsistent(Some(t)) =>
      display("Ticket reserved. Enter payment info.")
      computePrice(remaining) <- type error, using inconsistent value
    case Inconsistent(None) =>
      display("Try again.")
    case Consistent(None) =>
      display("Sorry, all sold out.")
  }
}
```