Real-Time Course

Transaction based temporal model for Real-time databases

Peter van der Stok@philips.com
TU/e Computer Science, System Architecture and Networking

Real-time data

Data used in classical administration system
"bank account"
-> represent status of constant real-world

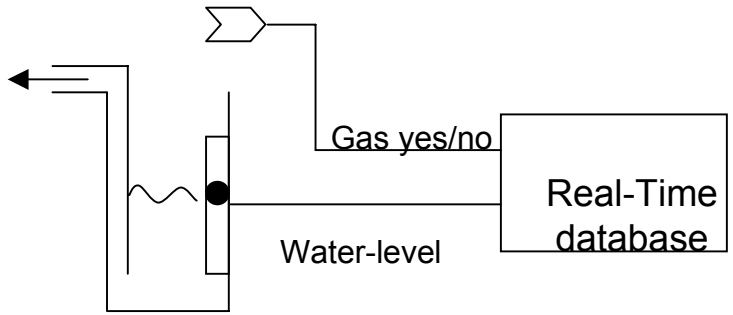
Data used in real-time systems

"current position"	(flight control system)
"outside temperature"	(home heating system)
"water level"	(mine pumps)
"stock price PHILIPS"	(stock market investor)

-> represent status of changing world

Normal consistency criteria do not suffice
-> the database has to "refresh" its content to accurately represent the real world

Temporal data

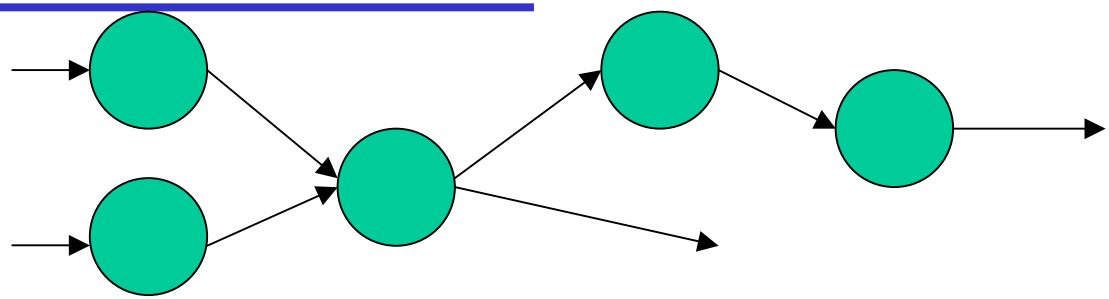


Distinguish “temporal” and “non-temporal” data

New consistency criteria?

-data items equal status of the world
-> impossible to realize, system is not instantaneous
-data items equal status of the world within error margin
-> impossible without notion of distance
-> impossible without bounded change rate
- data items follow the change of the real-world within bounded time

Temporal model



Temporal item X has an absolute validity duration AVD(X)
-> transactions never read values of X older than AVD(X)

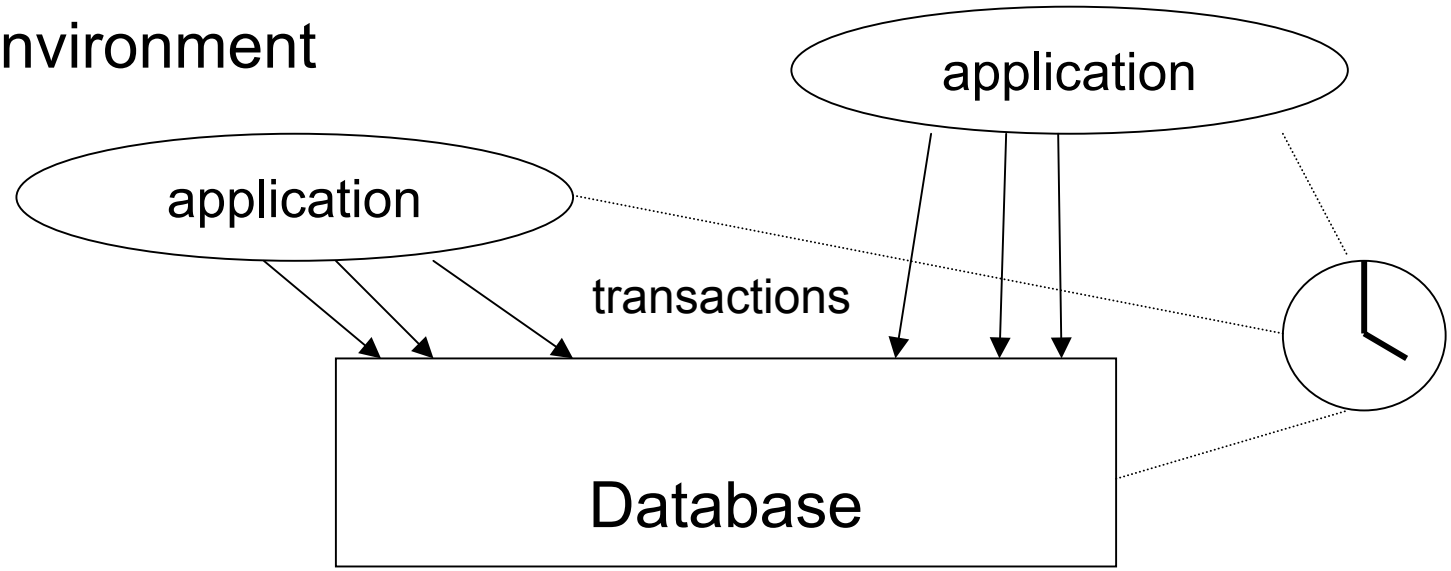
Example

Suppose the mine-pump system has to react to a change in water-level in 10 seconds

Suppose AVD(water-level) = 5

The reaction should take place within 10 –5 = 5 seconds

Temporal model



- Applications issue real-time transactions to database
- Global clock τ_{now} used by the database
- Applications specify real-time demands using τ_{now}

Temporal model

Database contains set of data-items
Data-items can be atomically *read* and *written*
A data-item X is represented with an instance x

An instance x is unique and has a value x.v and some control information (defined later)

When X is written a new instance x is created and old instance is removed

Two types of data-items
- non-temporal items
- temporal items

Temporal model

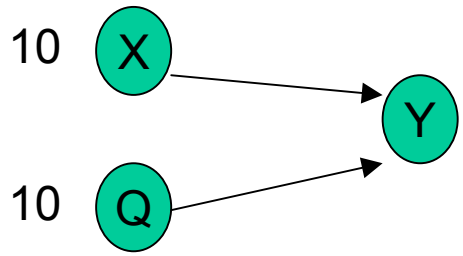
Trivial: as time progresses instances grow old.
The *time of measurement* x.τ of instance x defines the moment that **x.v** accurately reflected the real world
Age of instance **x** is τ_{now} **- x.τ**

Two types of temporal items

- sensor items

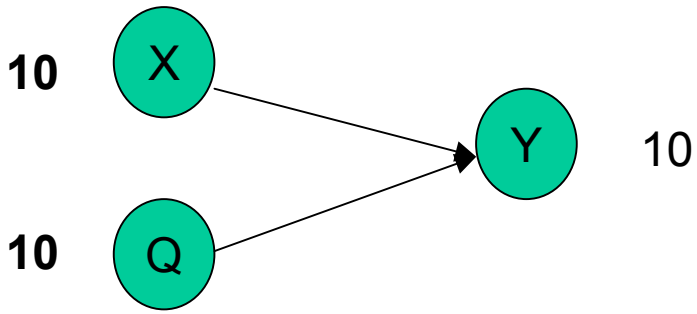
  x.τ is defined by transaction that writes x
- derived items

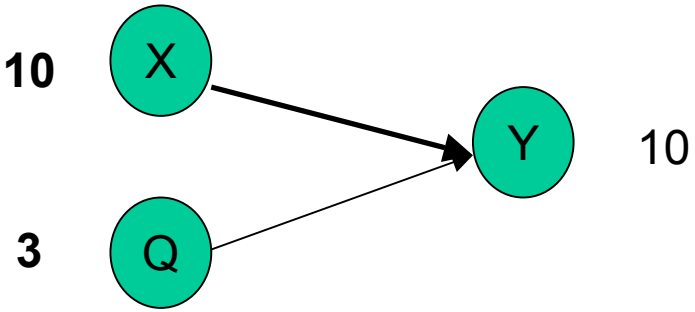
  these items were never measured



Derivation rules

Relation with the time
of measurement of input





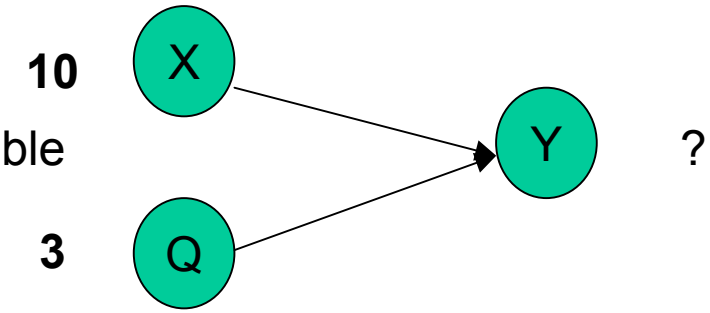
Semantic information about derivation
Is available
X is time critical
Q is not time critical

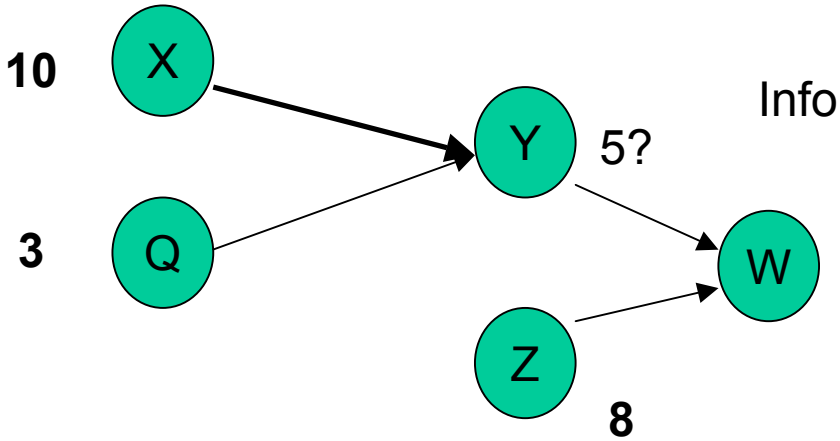
Derivation rules

No semantic information available

X ?

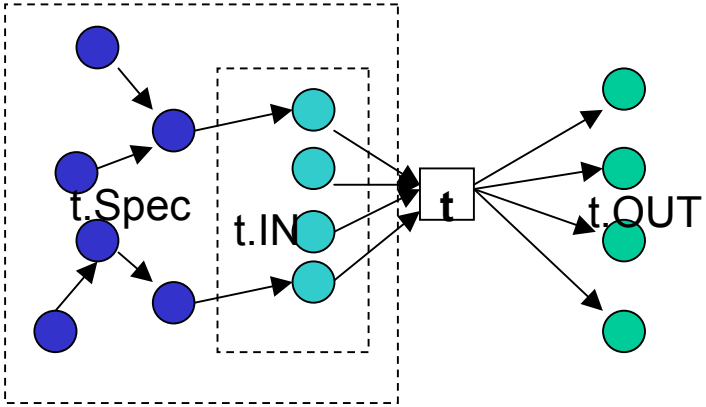
Q ?





Information hiding in derivation trees

Temporal sets



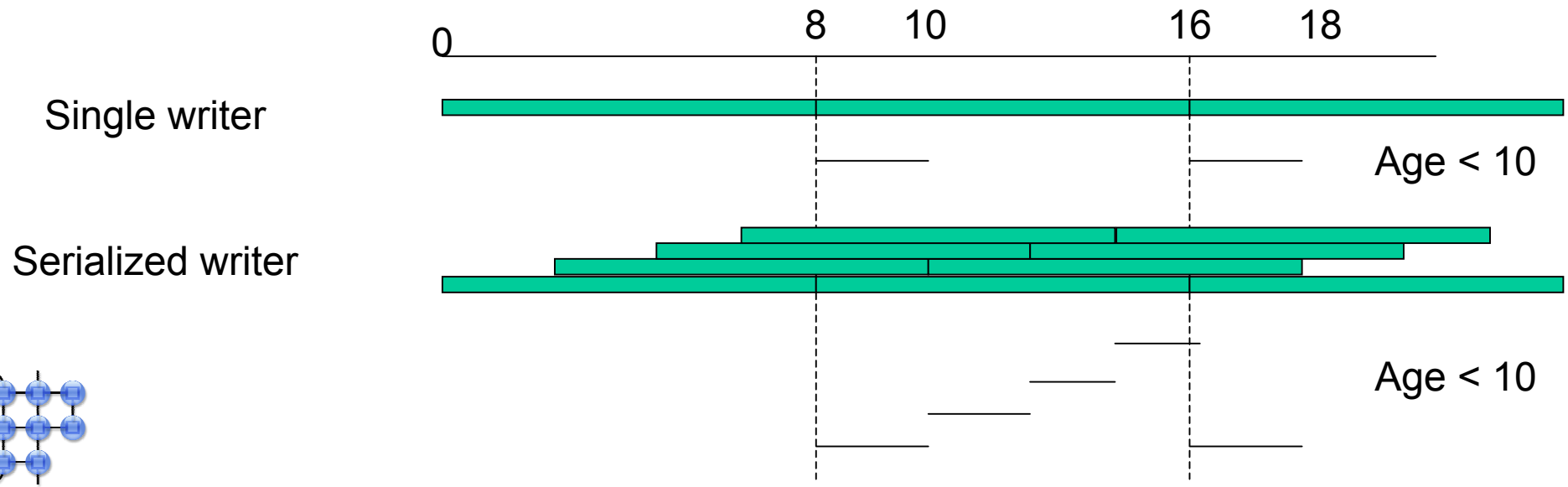
t specifies temporal constraints
for temporal items in t.Spec

t.IN set of temporal items that t reads
t.OUT set of temporal items that t writes
t.Spec set of temporal items, constrained by t

t specifies maximal age *t.m*(X) for each item X in t.IN
t specifies maximal age *t.m*(Q) for each item Q used to derive X in t.IN

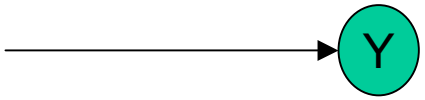
Transaction types

- **User transactions** do not write temporal data t.OUT =ø
  have effects outside database or write non-temporal data
- **Refresh transactions** have a supporting role

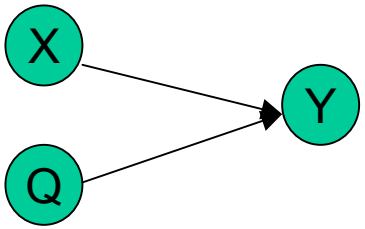


Refresh Transactions

- **Sensor transactions** t.IN =ø



- **Derive transactions**



Absolute consistency

Database *absolutely temporally consistent* if
- all transactions satisfy all temporal constraints

Forall **t**: forall **X** in **t.Spec**
t reads an instance **x** with τ_{now} **- x.τ <= t.m(X)**

Requirement holds at start of transaction

Peter van der Stok@philips.com
TU/e Computer Science, System Architecture and Networking

Relative consistency

For each **t,** a set *t.rtc* of relative temporal constraints, *rc* is specified.

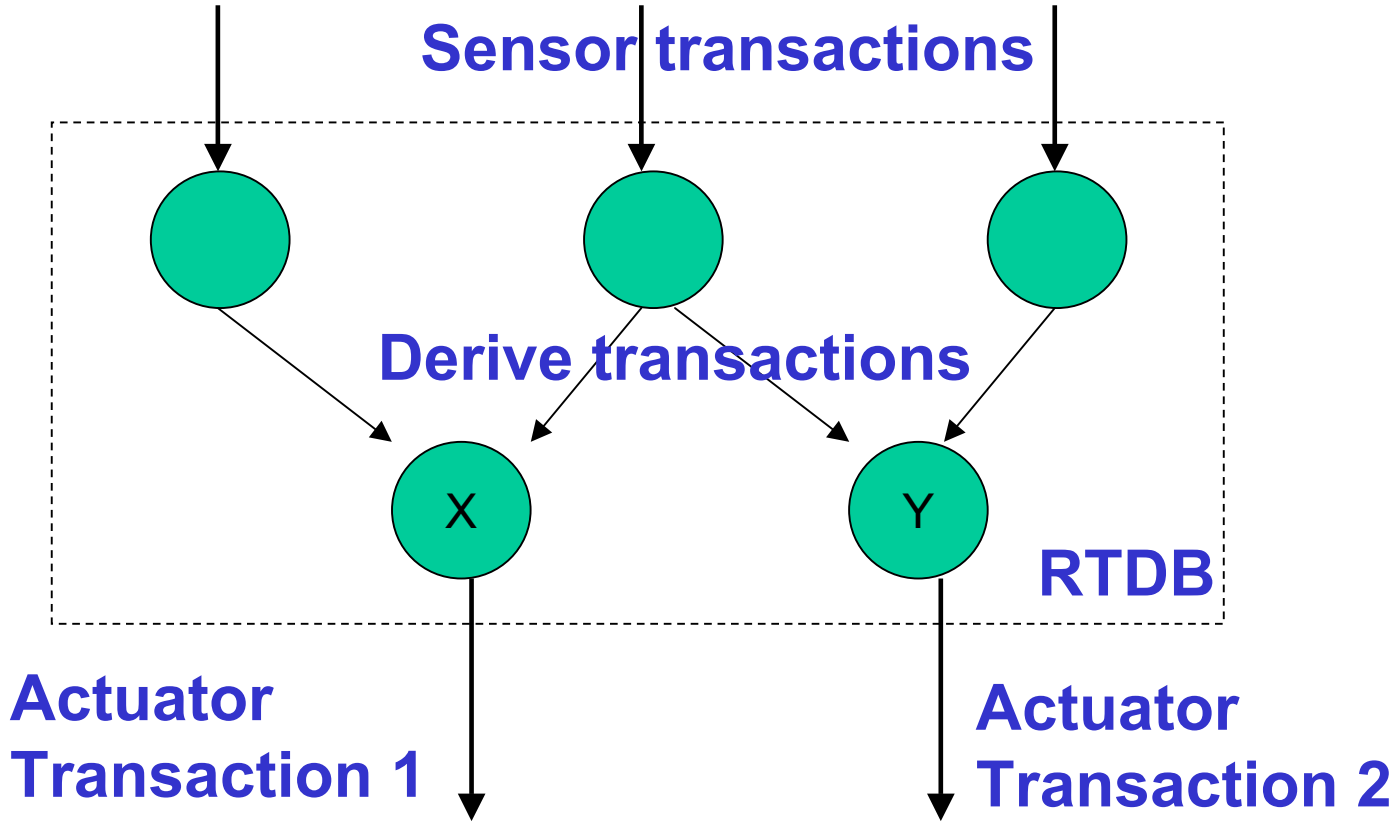
Constraint *rc* consists of:

- a subset *rc.s* in **t.Spec** of data-items
- a maximal age <u>difference</u> *rc.m*

Instances read by **t** must have t.o.m such that
Forall X,Y in *rc.s:* | **x.τ - y.τ** | <= *rc.m*

If all *rc* of all transactions are satisfied, the database is *<u>relatively</u> temporally consistent*

Database and transactions



TOM functions

- Sensor transaction: worst case assumption

  $tom(\mathbf{t,x}) = \text{start}(\mathbf{t})$
- Derive transaction: worst case assumption

  $tom(\mathbf{t,y}) = min_{x\ in\ t.in}\ \mathbf{x.\tau}$
- One critical input X

  $tom(\mathbf{t,y}) = \mathbf{x.\tau}$
- Proportional influence

  $tom(\mathbf{t,y}) = \Sigma_{x\ in\ t.in}\ \mathbf{x.\tau}/|\mathbf{t}.in|$
- Prediction model

  transaction t uses instance **x** to create instance **y** *which* is " measured" in the future: $\mathbf{y.\tau} > \tau_{now}$

  $tom(\mathbf{t,y}) = \mathbf{x.\tau} + \delta$

Design decisions

- **First decision**: all requirements are met at all times

-> a maximal age ma_X exists for each X

$$ma_X = \min(\min_{t,X:\ X \text{ in } t.Spec} t.ma(X),\ \min_{t:\ rc \text{ in } t.rtc \text{ and } X \text{ in } rc.s} rc.m)$$

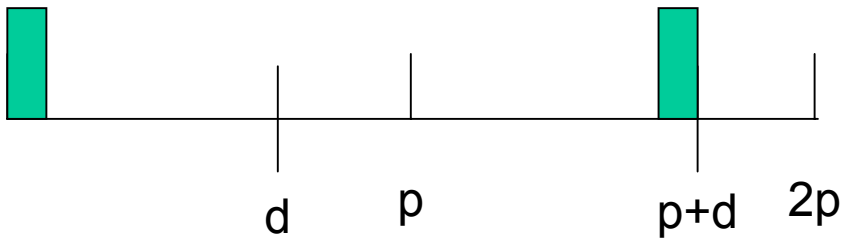
- **Second decision**: refresh transactions are periodic

- **Sensor transaction:**

  if **t** writes X, then t.p + t.d <= ma_X

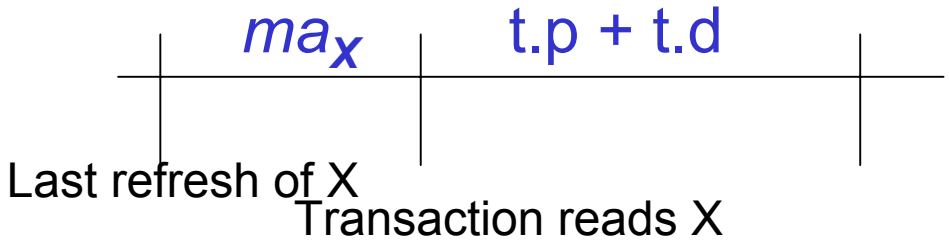
t.p is period of **t**

t.d is deadline of **t**



Derived age calculation

Derive transactions read instances that have aged worst case:
Reaction time system is
ma_X + t.p + t.d



Additional age aa_Y is maximum of *t.start- tom(t,y)*

Example **t** reads X and Q to write Y
tom(t,y) = 0.5 **x.τ** + 0.5 **q.τ**
If ma_X in [0,8] and ma_Y in [0,6] then aa_Y = 7

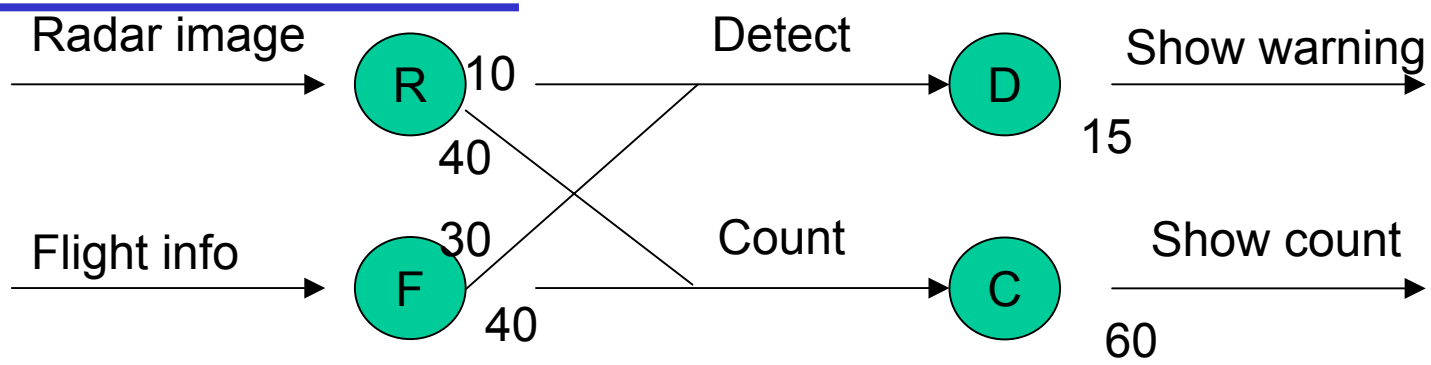
t.p + t.d <= ma_Y - aa_Y

Derived age calculation(2)

This holds for each Y that transaction **t** writes

$$t.p + t.d <= \min(Y \text{ in } t.OUT: ma_Y - aa_Y)$$

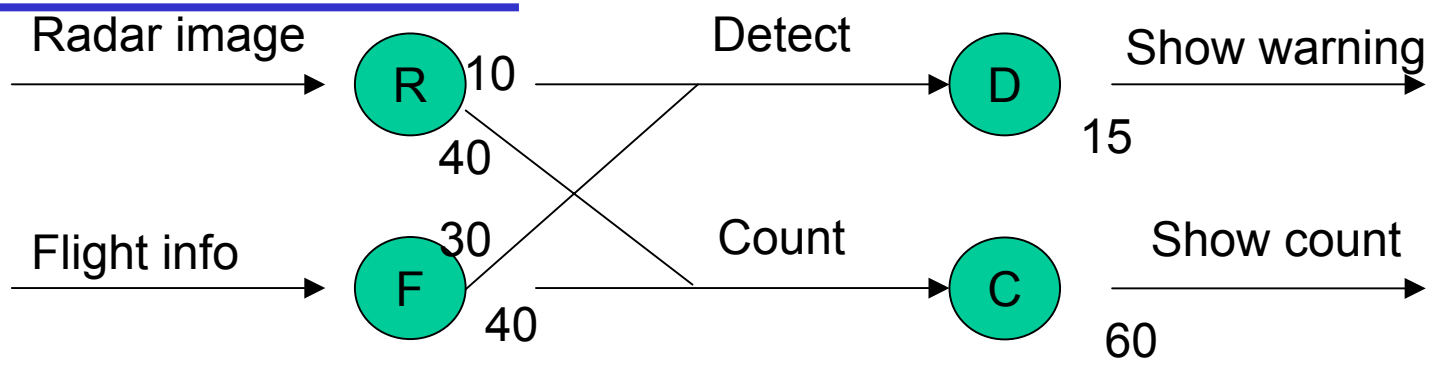
Example



Show Warning: SW.p=15 and SW.m(D)=15
Show Count: SC.p=60 and SC.m(C)=60

tom(Detect,D) = **r.τ** tom(Count,C) = min(**r.τ**, **f.τ**)
tom(Radar image, R) = Radar image.start
tom(Flight info, F) = Flight info.start

Example (2)

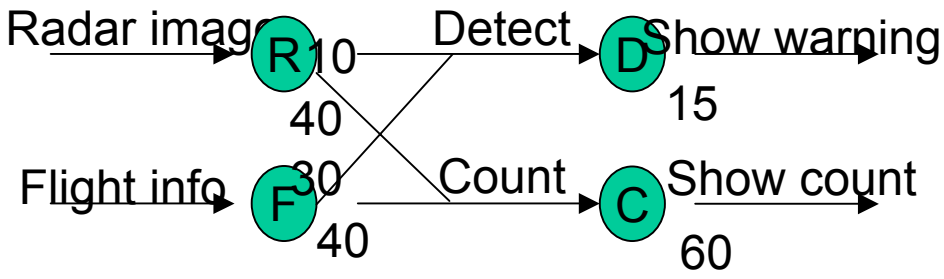


Design choice requirements on detect and count

Detect.m(R) = 10
rc.s = {R,F} with rc.m=30 and rc in detect.rtc
Count.m(R) = 40
Count.m(F) = 40

Example (3)

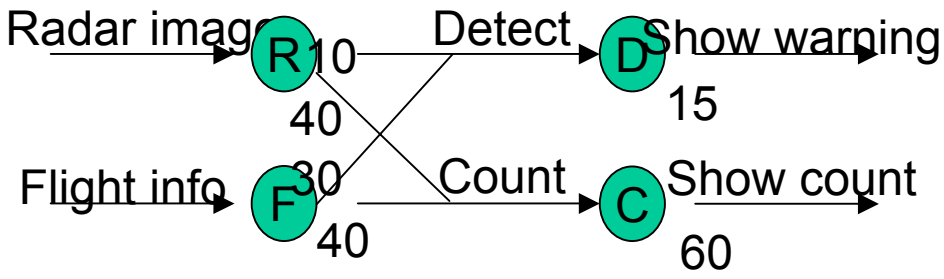
Detect.m(R) = 10
Count.m(R) = 40
Count.m(F) = 40
rc.s = {R,F}
rc.m=30



ma_D = SW.m(D) = 15
ma_C = SC.m(C) = 60
ma_R = min(Count.m(R), Detect.m(R), rc.m) = 10
ma_F = min(Count.m(F), rc.m) = 30
tom(Detect,D) = **r.τ** , ma_R = 10 ->aa_D =10
tom(Count,C) = min(**r.τ, f.τ**) , ma_R = 10, ma_F = 30 ->aa_C =30

Example (4)

$ma_D = 15$, $aa_D = 10$
$ma_C = 60$, $aa_C = 30$
$ma_R = 10$
$ma_F = 30$



detect.p + detect.d <= $ma_D - aa_D = 5$
count.p + count.d <= $ma_C - aa_C = 30$
radar image.p + radar image.d <= $ma_R = 10$
Flight info.p + flight info.d <= $ma_F = 30$