Coloured Petri Net Example

Vincent J. Maccio

Department of Computing and Software
McMaster University
Hamilton, Ontario, Canada

The Bank Courier Problem

- A set of trucks drives around the city collecting cheques and mail
- Locations are hubs, branches, main, processing centre
- Model using a CPN, built up from SADT models

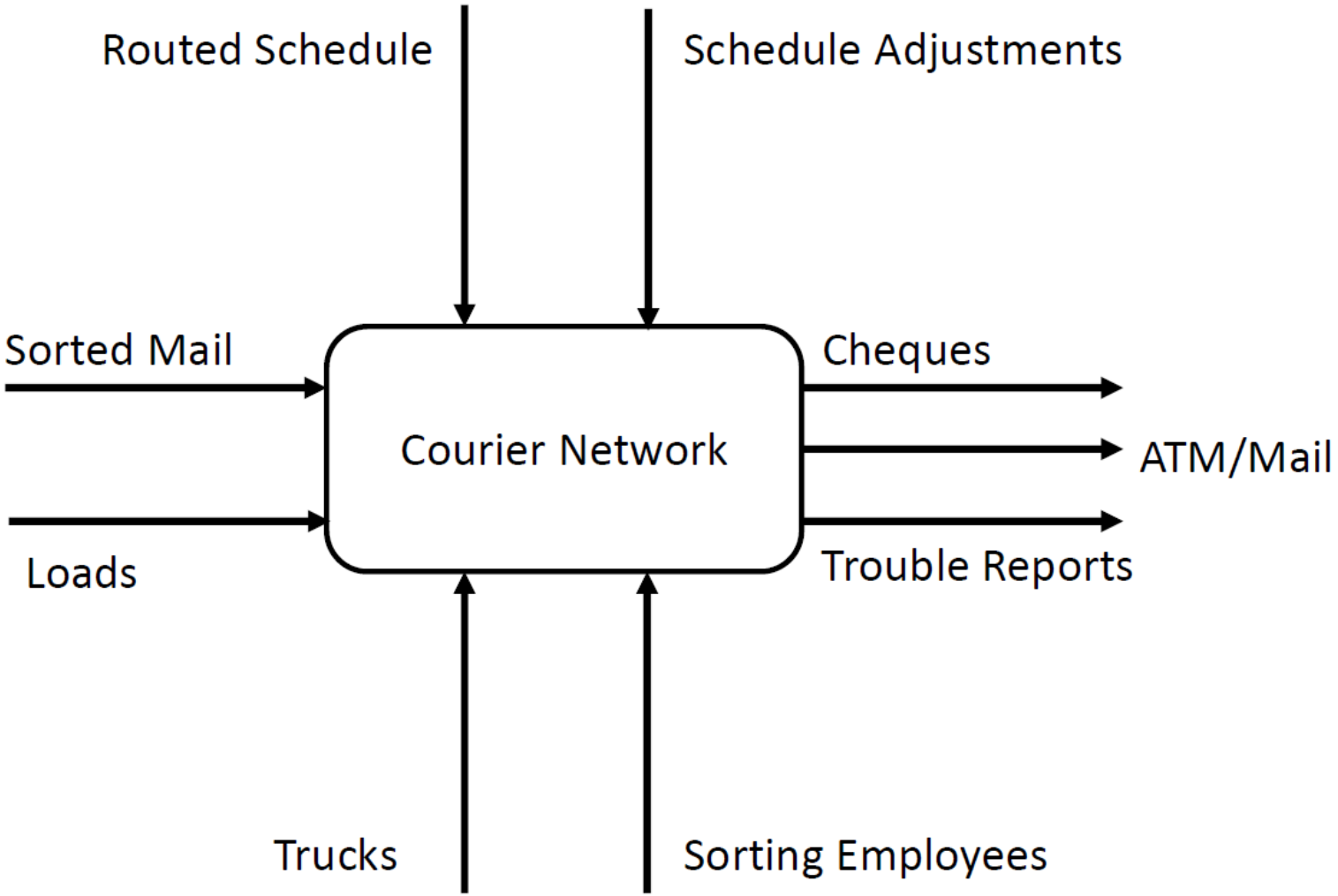
Data Format

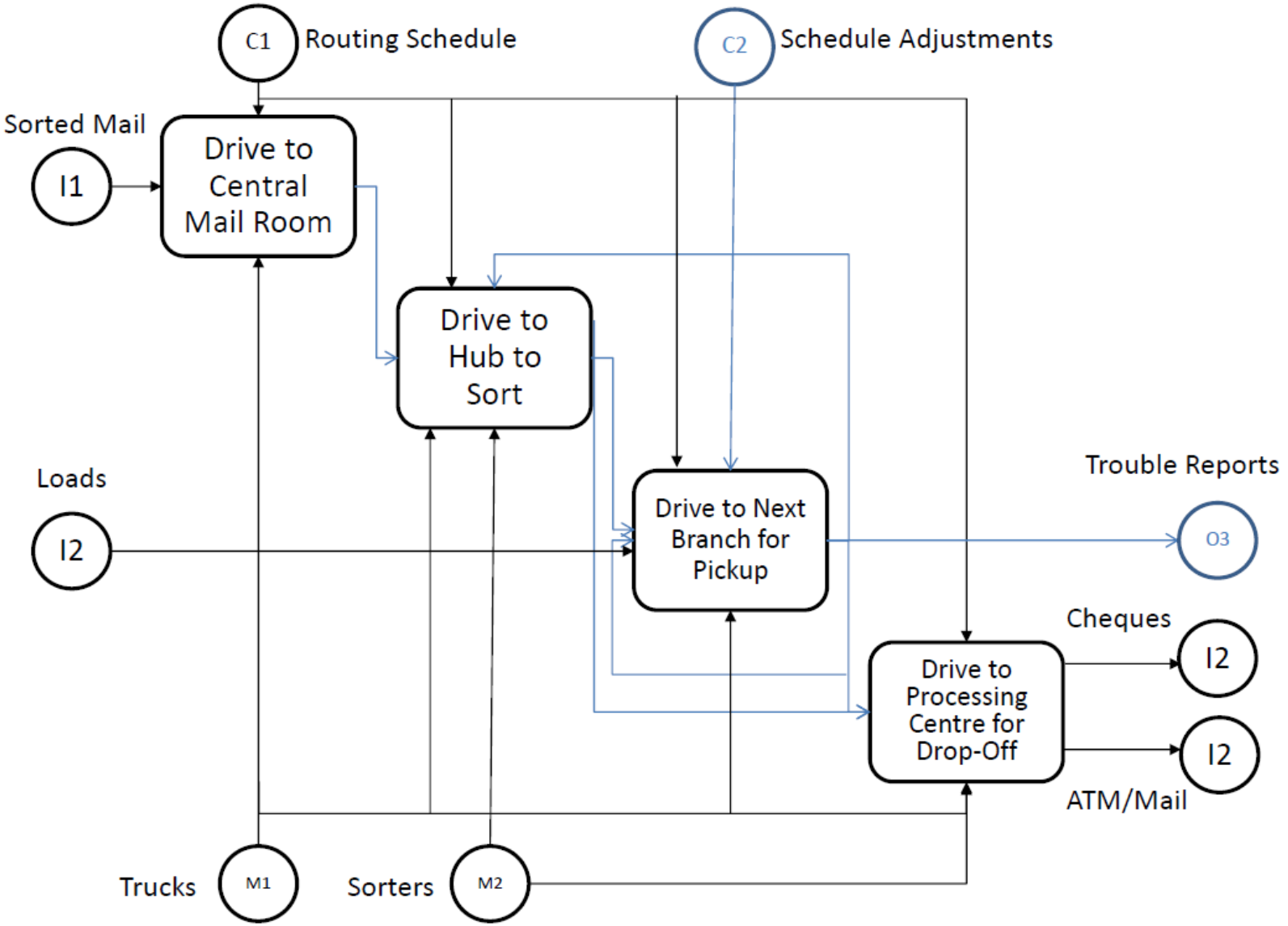
Route	Time	Type	Location
4	7:45	Start	Boston
	8:20	Branch	B483
	9:10	Branch	B488
	.....	.....	.....

Branch	Time	Cheques	Mail	ATM
B483	8:20	360	No	Yes
	12:10	620	Yes	No
	.....	.....	.....	.....

Structured Analysis and Design Technique (SADT)

- Describe complex systems as a series of functions
- A variable can be one of four:
  - Inputs represent data or consumables that are needed by the activity
  - Outputs represent data or products that are produced by the activity
  - Controls represent commands which influence the execution of an activity but are not consumed
  - Mechanisms identify the means, components or tools used to accomplish the activity


Routed Schedule
Schedule Adjustments
Sorted Mail
Courier Network
Cheques
ATM/Mail
Loads
Trouble Reports
Trucks
Sorting Employees


C1
Routing Schedule
C2
Schedule Adjustments
Sorted Mail
I1
Drive to Central Mail Room
Drive to Hub to Sort
Loads
I2
Drive to Next Branch for Pickup
Trouble Reports
O3
Cheques
I2
Drive to Processing Centre for Drop-Off
I2
ATM/Mail
Trucks
M1
Sorters
M2

The Colours

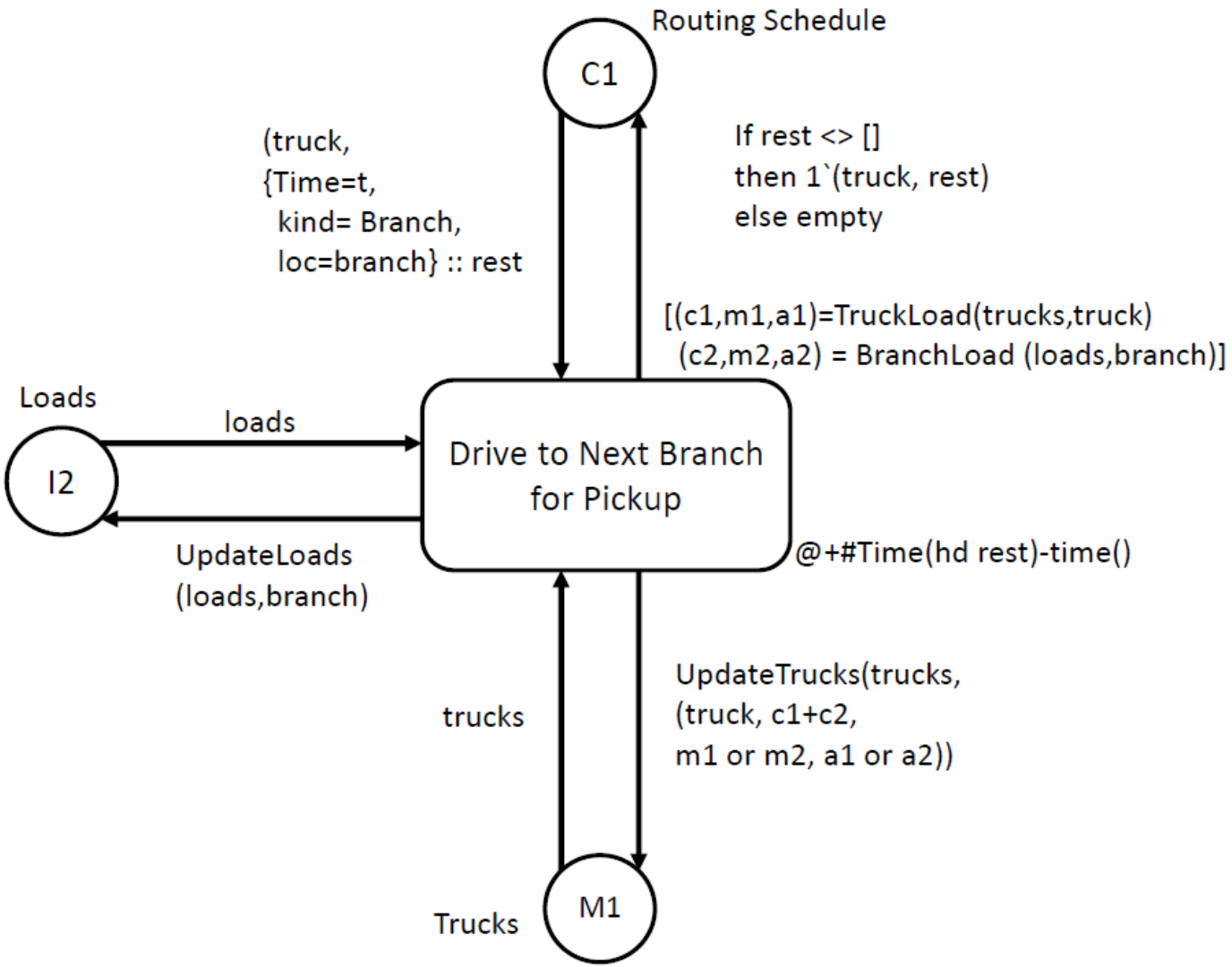
```
colour Kind = with ToMain | Hub | Branch | FromMain | Start;
colour Location = string;
colour Stop = rcord Time:TIME * kind:Kind * loc:Location;
colour Route = list Stop;
colour TruckId = int;
colour RouteSchedule = product TruckId * Route timed;
colour Cheques = int;
colour Mail = bool;
colour ATM = bool;
colour Truck = product TruckId * Cheques * Mail * ATM;
colour Trucks = list Truck
colour Load = product Location * Cheques * Mail * ATM;
colour Loads = list Load;
```

Colour Example

Route	Time	Type	Location
4	7:45	Start	Boston
	8:20	Branch	B483
	9:10	Branch	B488
	.....	.....	.....

Becomes...

(4, [{time=745, kind=Start, loc="Boston" },
{time=820, kind=Start, loc="B483" },
{time=910, kind=Start, loc="B488" }.....])@ 745


Routing Schedule
C1
(truck,
{Time=t,
kind= Branch,
loc=branch} :: rest
If rest <> []
then 1`(truck, rest)
else empty
[(c1,m1,a1)=TruckLoad(trucks,truck)
(c2,m2,a2) = BranchLoad (loads,branch)]
Loads
loads
I2
Drive to Next Branch
for Pickup
UpdateLoads
(loads,branch)
@+#Time(hd rest)-time()
trucks
UpdateTrucks(trucks,
(truck, c1+c2,
m1 or m2, a1 or a2))
Trucks
M1

Conclusion

- An interesting design process for tackling problems
- The project was a success on the small scale
- However, the implications have wider bearing on a larger scale