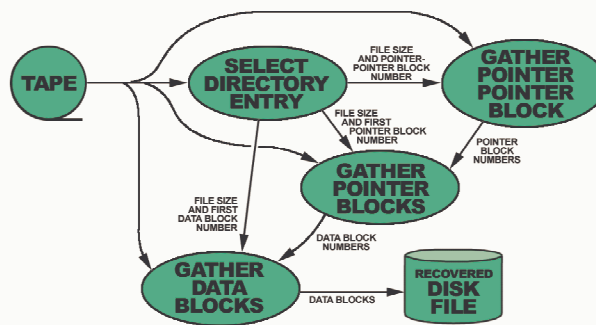


E-Genting Programming Competition 2004

Pre-Competition Workshop, Week 4
12 October 2004

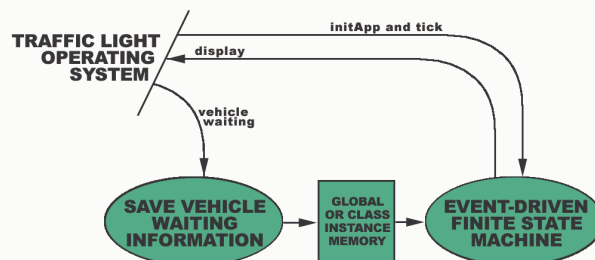
Dataflow Revisited



DATA FLOW DIAGRAM

Process Implementations

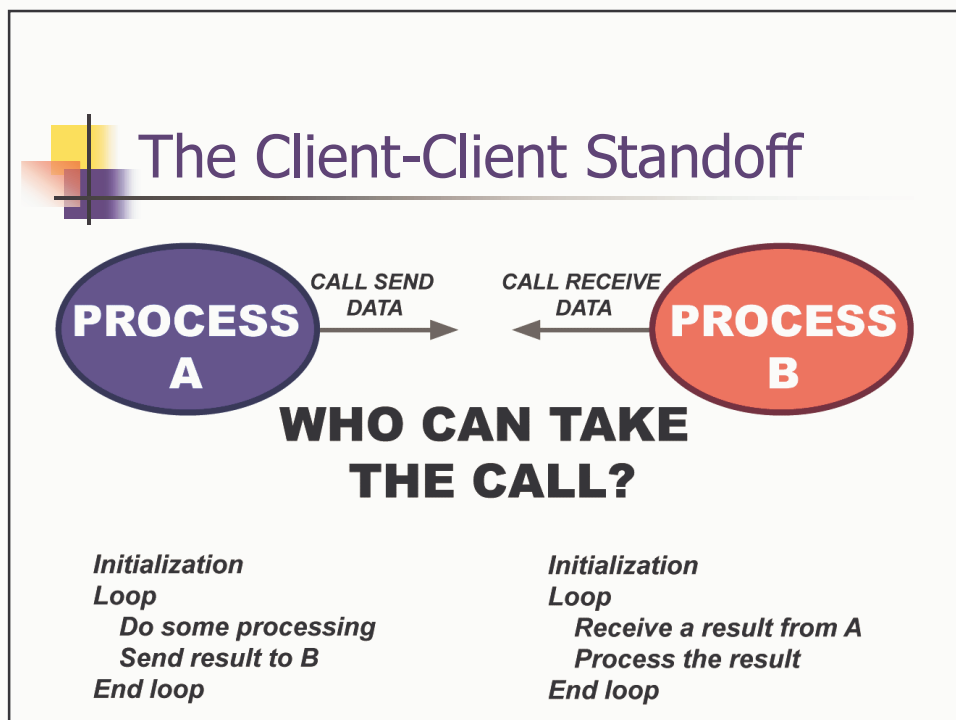
- a program;
- a thread;
- a class;
- a function;
- an interrupt service routine or a group of interrupt service routines;
- in-line code;
- shared memory;
- a pipe.



Event-driven finite state machine:

1. Wait for initiation ['UNINITIATED' state];
2. branch = NORTH;
3. Loop:
 - a) Display green light combination (branch);
 - b) For green time (branch):
 - i. Wait for a clock tick ['GREEN' state];
 - c) Display amber light combination (branch);
 - d) For amber time (branch):
 - i. Wait for a clock tick ['AMBER' state];
 - e) Read vehicle waiting information;
 - f) branch = Next (branch, vehicle waiting information);
4. End loop.

Event	State	Procedure
Initiate	UNINITIATED	<ol style="list-style-type: none"> 1. branch = NORTH; 2. Display green light combination (branch); 3. state = GREEN;
Clock tick	GREEN	<ol style="list-style-type: none"> 1. If green time(branch) has elapsed: <ol style="list-style-type: none"> a. Display amber light combination (branch); b. state = AMBER; 2. End if.
Clock tick	AMBER	<ol style="list-style-type: none"> 1. If amber time(branch) has elapsed: <ol style="list-style-type: none"> a. Read vehicle waiting information; b. branch = Next (branch, vehicle waiting information); c. Display green light combination (branch); d. state = GREEN; 2. End if.





Process B as a Finite State Machine

Stimulus	State	Processing
Initiate	UNINITIATED	1. Initialisation; 2. state = READY.
Result	READY	1. Process the result.



With a Little Optimisation

1. Initialisation of Process A.
2. Initialisation of Process B.
3. Loop:
 - a. Do the processing from Process A;
 - b. Process the result as per Process B;
4. End loop.