Composite and Visitor Pattern
Computer structure

• Every part has price
• the price of a component part is the sum of all its (recursive) sub parts
Component

defered class
    COMPUTER_COMPONENT

feature -- Access

    price: INTEGER is
        -- Price of this computer part
        deferred
        ensure
            good_result: Result >= 0
        end
    end

end
Composite

class
    COMPUTER_COMPOSITE
inherit
    COMPUTER_COMPONENT

feature -- Access
    parts: LINKED_LIST [COMPUTER_COMPONENT]
        -- Parts in Current composite

    price: INTEGER is
        -- Price of this composite part
    do
        from
        parts.start
        until
        parts.after
        loop
            Result := Result + parts.item.price
        end parts.forth
    end
end
end
CPU

class
  CPU
inherit
  COMPUTER_COMPONENT
  redefine price end

feature -- Access

  price: INTEGER
    -- Price of this computer part

feature -- Operations

  perform_addition is ...

  perform_substraction is ...

end
Monitor

class MONITOR

inherit COMPUTER_COMPONENT
   redefine price end

feature -- Access

   price: INTEGER
      -- Price of this computer part

feature - Operations

   display is ...

end
Assemble a computer

```python
graph_card: COMPUTER_COMPOSITE
main_memory, graph_memory: MEMORY

cpu: CPU
gpu: GPU
monitor: MONITOR
computer: COMPUTER_COMPOSITE
...

-- Initialize graph card.
graph_card.parts.extend(gpu)
graph_card.parts.extend(graph_memory)

-- Initialize computer.
computer.parts.extend(cpu)
computer.parts.extend(main_memory)
computer.parts.extend(monitor)
computer.parts.extend(graph_card)

-- Print price.
print(computer.price)
```

What if we just want price of memories?
What if we just want price of CPU or GPU?
What if ...
Visitor Pattern

• Process the elements of an (unbounded) data structure.

• Apply computations depending on the type of data node.

• Store the code outside of the data structure.

• Most of the time used together with the composite pattern.
deferred class
  COMPUTER_VISITOR

feature -- Process

  process_cpu (a_cpu: CPU) is deferred end
  -- Process a_cpu.

  process_monitor (a_monitor: MONITOR) is deferred end
  -- Process a_monitor.

  process_memory (a_memory: MEMORY) is deferred end
  -- Process a_memory.

  process_gpu (a_gpu: GPU) is deferred end
  -- Process a_gpu.
process_composite (a_composite: COMPUTER_COMPOSITE) is
  -- Process a_composite.
  do
    from
    a_composite.parts.start
  until
    a_composite.parts.after
  loop
    a_composite.parts.item.process (Current)
    a_composite.parts.forth
  end
end
end
Component

defered class
  COMPUTER_COMPONENT

feature -- Process

  process (a_visitor: COMPUTER_VISITOR) is
    -- Process Current using a_visitor.
    deferred
    end
  end

end
class CPU
inherit COMPUTER_COMPONENT

feature -- Access

  price: INTEGER
    -- Price of this computer part

feature -- Operations

  perform_addition is ...
  perform_substraction is ...

feature -- Process

  process (a_visitor: COMPUTER_VISITOR) is
    -- Process Current using a_visitor.
    do
      a_visitor.process_cpu (Current)
    end

end
class MEMORY
inherit COMPUTER_COMPONENT

feature -- Access

  price: INTEGER
    -- Price of this computer part

feature -- Operations
  display is ...

feature -- Process

  process (a_visitor: COMPUTER_VISITOR) is
    -- Process Current using a_visitor.
    do
      a_visitor.process_memory (Current)
    end

end
Composite

class COMPUTER_COMPOSITE
inhibit COMPUTER_COMPONENT

feature -- Access

  parts: LINKED_LIST [COMPUTER_COMPONENT]
     -- Parts in Current composite

feature -- Process

  process (a_visitor: COMPUTER_VISITOR) is
     -- Process Current using a_visitor.
     do
     a_visitor.process_composite (Current)
   end

end
MEMORY_PRICE_VISITOR (1/2)

class MEMORY_PRICE_VISITOR
  inherit COMPUTER_VISITOR

feature -- Calculate

  memory_price (a_component: COMPUTER_COMPONENT): INTEGER is
    -- Price of all kinds of memory in a_component
    do
      last_price := 0
      a_component.process (Current)
    end

feature -- Access

  last_price: INTEGER
    -- Last calculated price
feature -- Process

process_memory (a_memory: MEMORY) is
    do
        last_price := last_price + a_memory.price
    end

process_cpu (a_cpu: CPU) is
    do
    end

process_monitor (a_monitor: MONITOR) is
    do
    end

process_gpu (a_gpu: GPU) is
    do
    end

end
Double Dispatch

\[\text{component} := \text{memory}\]
\[\text{visitor} := \text{memory\_price\_visitor}\]

...\[
\text{component}.\text{process} \ (\text{visitor})\]