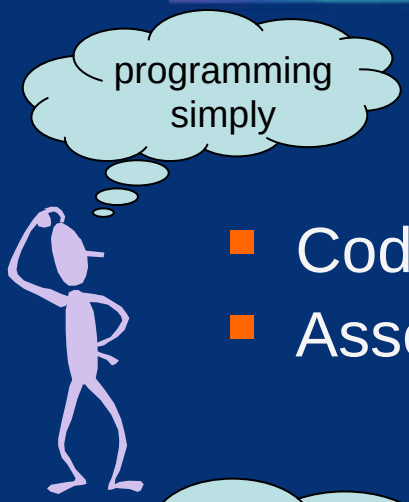
Towards Software Component Assembly Language Enhanced with Workflows and Skeletons

Marco ALDINUCCI and Marco DANELUTTO
UNIPI - University of Pisa (Italy)
Hinde Lilia BOUZIANE and Christian PEREZ
Projet-team GRAAL - INRIA Rhône-Alpes - ENS Lyon (France)

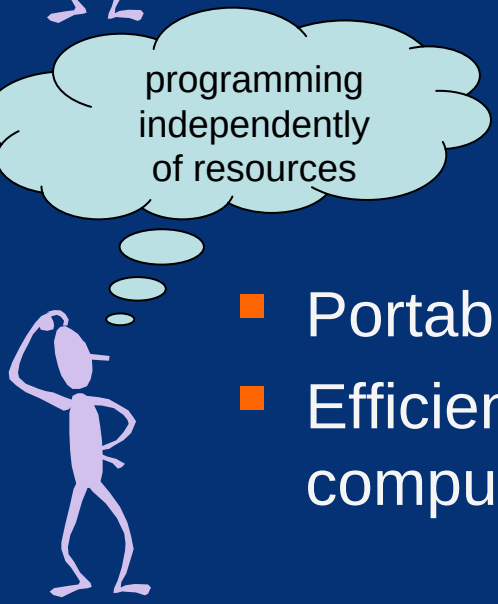
Outline of the talk

- Introduction
  - Software component models
- Existing models
  - STCM: a spatio-temporal component model
  - Skeletons based models for parallel programming
- STKM: proposal of skeletons introduction in STCM
  - Objectives
  - Overview
  - Example of usage
- Conclusions and future works

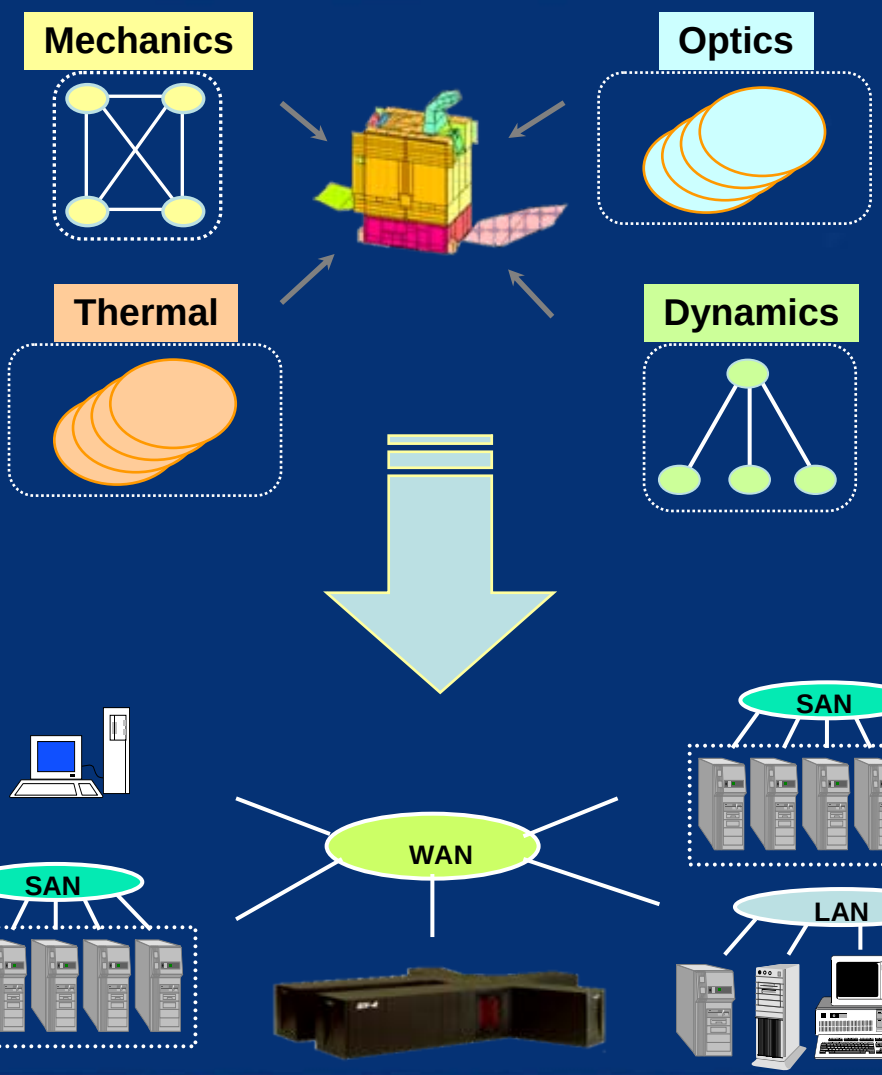
Context and problematic



- Code reuse
- Assembly



- Portability
- Efficient computation

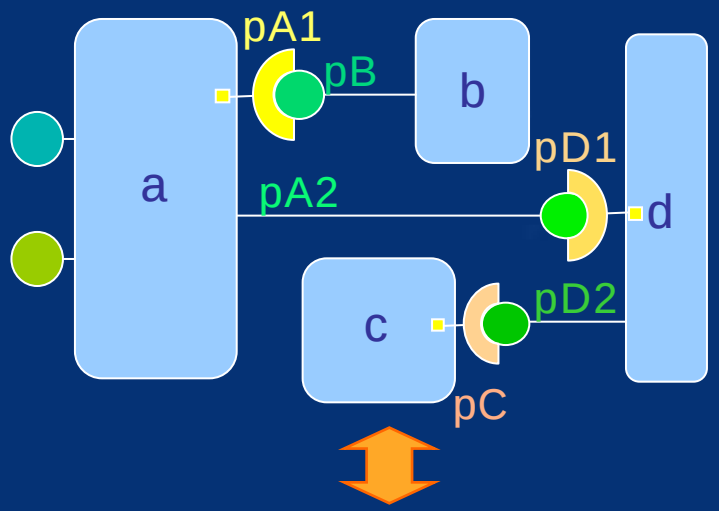


Software component (1/2)

- Black box
- Ports
  - Method invocation (CCM, CCA, Fractal/GCM, SCA)
  - Events (CCM)
  - Streams (Grid.it/ ASSIST)
  - Message passing (Darwin)
  - Document passing (SCA)
  - …

Assembly

- Component instances and connections
- Architecture Description Language (ADL)
  - CCM, GCM, …
- Dynamic (API)
  - CCA, CCM, GCM, ...



ADL

instanceComp:
a: A, b: B, c: C, d: D;
connections:
a.pA1 <-> b.pB;
c.pC <-> d.pD2;
d.pD1 <-> a.pA2;

Limitations of existing component models

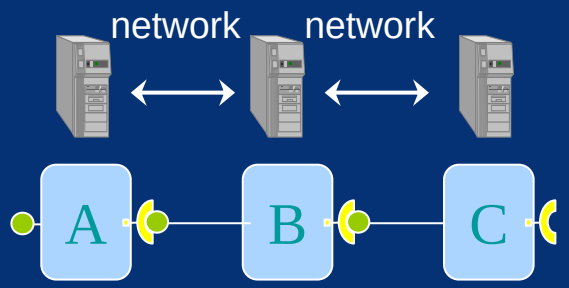
- Assembly models close to the computing resources
  - Behavior hidden in the assembly

- “Over-consumption” of resources

➡ **Workflow models**

  - Simple spatial relations

- Resource dependencies
- Complex design
  - Parallel paradigms (e.g. master-worker)



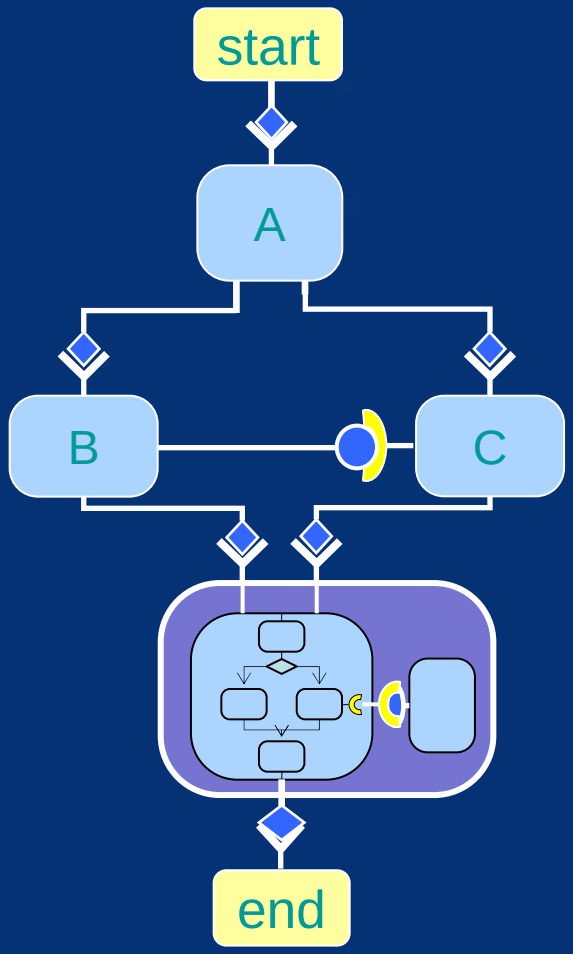
➡ **Algorithmic skeleton models**

Outline of the talk

- Introduction
  - Software component models
- Existing models
  - STCM: a spatio-temporal component model
  - Skeletons based models for parallel programming
- STKM: proposal of skeletons introduction in STCM
  - Objectives
  - Overview
  - Example of usage
- Conclusions and future works

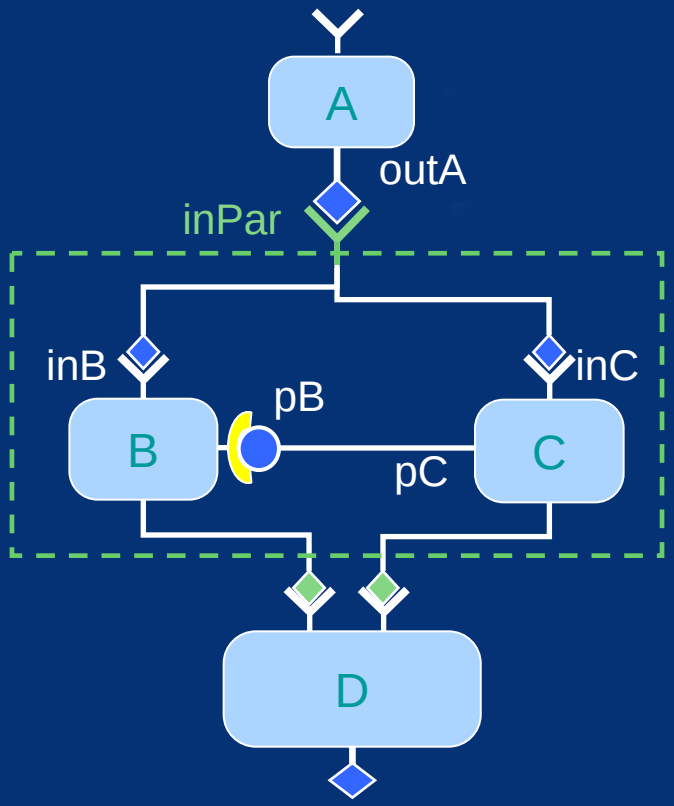
Overview of STCM [EuroPar'08]

- Combination of component and workflow models
  - Spatial and temporal dimensions at the same level of assemblies
- Component-task
  - Spatial ports (classical ones)
  - Input and output ports (temporal)
  - Task
- Assembly model
  - Adaptation of a workflow language



Assembly model

```
component Example {
  …
  parallel parCtrl {
    dataIn Double inPar <= a.outA;
    component B { dataIn Double inB;
                  clientPort Compute pB;
    … };
    component C { dataIn Double inC;
                  serverPort Compute pC;
    … };
    instance B b;
    instance C c;
    connect b.inB  to parCtrl.inPar;
    connect c.inC to parCtrl.inPar;
    connect b.pB to c.pC;
   // instructions
    section : exectask (a);
    section : exectask (b);
  } // end parallel
  …
}
```

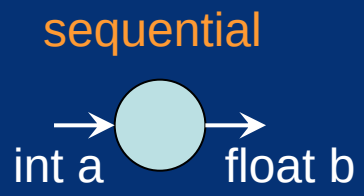


Algorithmic skeletons [M. Cole 1989]

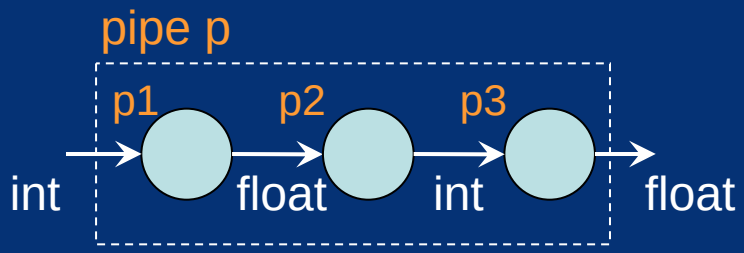
- Predefined patterns for parallel programming
  - Stream parallel
    - Pipeline, farm, …
  - Data parallel
    - Map (independent forAll), reduce, …
- Structured programming
  - Simplicity
  - Correctness of programs
- Hide the complexity of parallelism management
  - Creation of processes, data distribution, ..
- Behavioral skeletons add advanced management for adaptation

Algorithmic skeletons

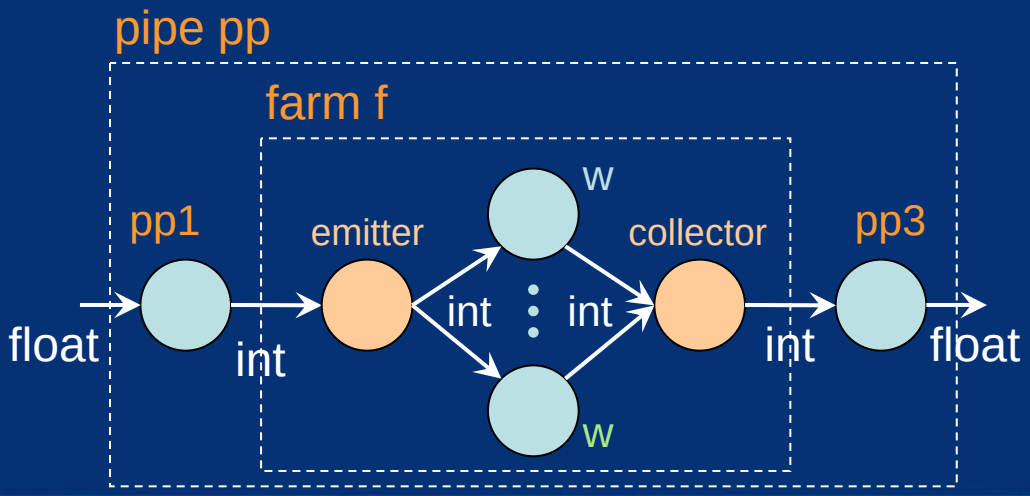
```
compute in (int a) out (float b)
      $ sequential code $
end
```




```
pipe p in (int a) out (float b)
       p1 in (a) out (float b1)
       p2 in (b1) out (int b2)
       p3 in (b2) out (b)
end pipe
```




```
farm f in (int af) out(int bf)
        w in (af) out(bf)
end farm
pipe pp in (float a) out(float b)
        pp1 in (b) out (int b1)
        f      in (b1) out (b2)
        pp3 in (b2) out (b)
end pipe
```



Outline of the talk

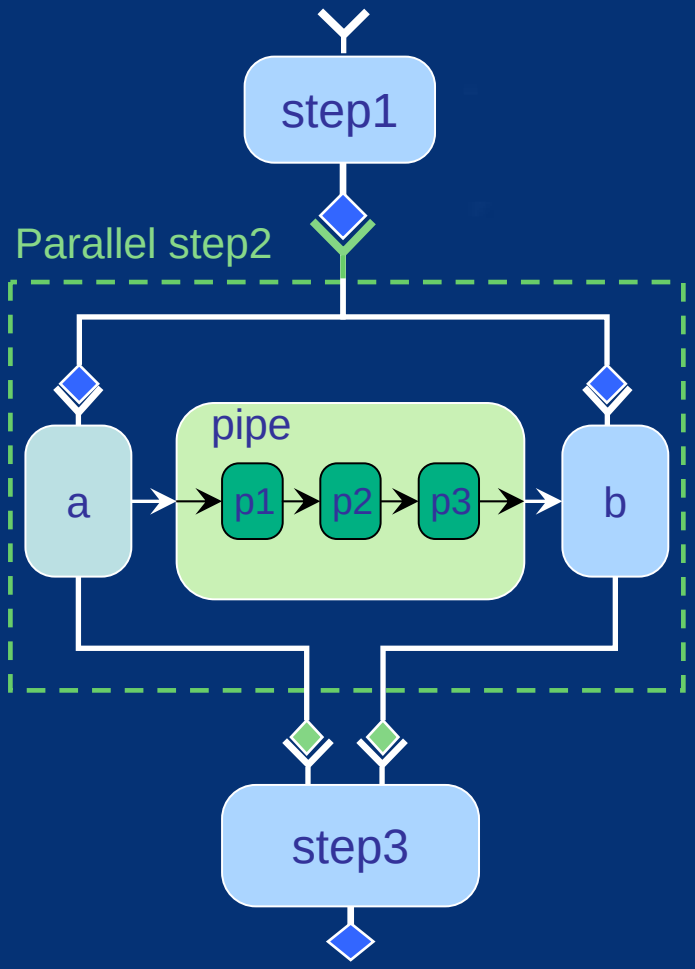
- Introduction
  - Software component models
- Existing models
  - STCM: a spatio-temporal component model
  - Skeletons based models for parallel programming
- STKM: proposal of skeletons introduction in STCM
  - Objectives
  - Overview
  - Example of usage
- Conclusions and future works

Objectives

- Simplifying programming parallel parts of an application
- Offering a similar level of abstraction as in skeleton models
- Portability on different execution resources
  - Code reuse
  - Efficiency

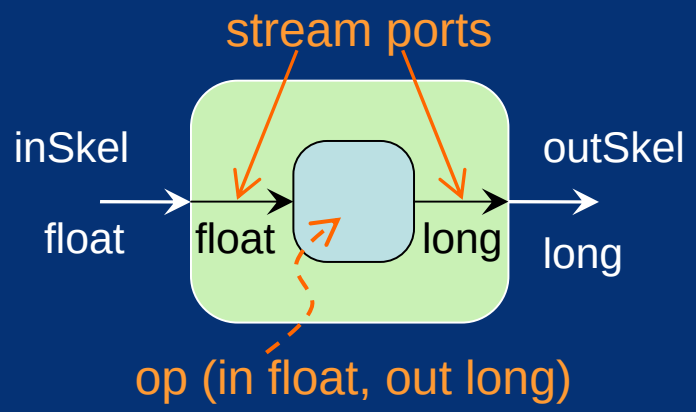
Overview of STKM

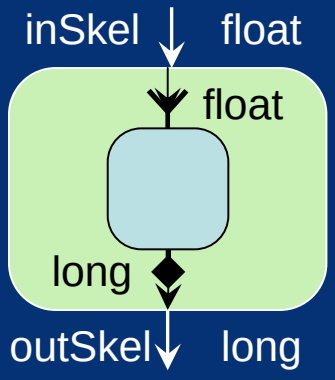
- Assembly model
  - STCM assembly + skeleton constructs
  - An STKM skeleton is a composite with a predefined behavior
  - Parameterization
    - Wrapping components
  - Usage in spatial and temporal dimension
    - Port cardinality principle (temporal dimension)



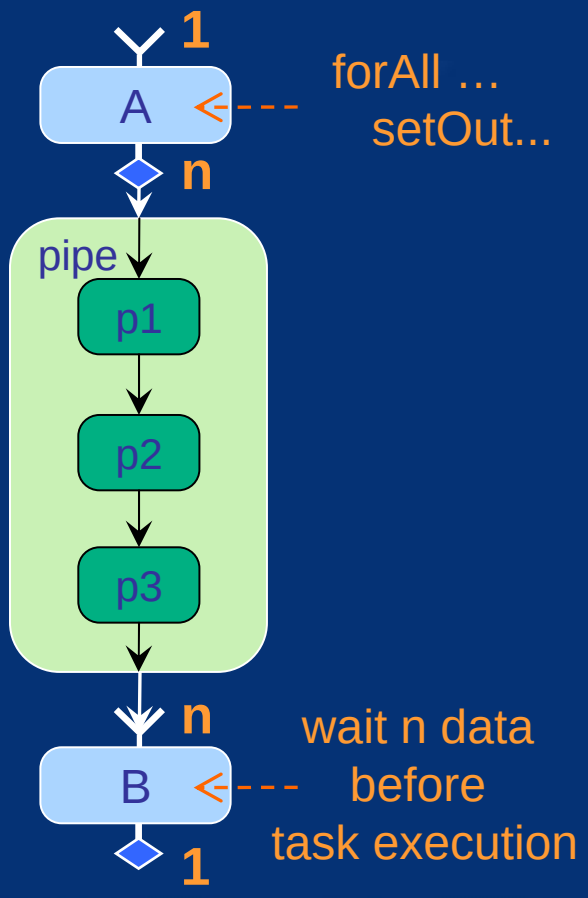
Component wrapping and port cardinality principle

- A skeleton element is a wrapped component



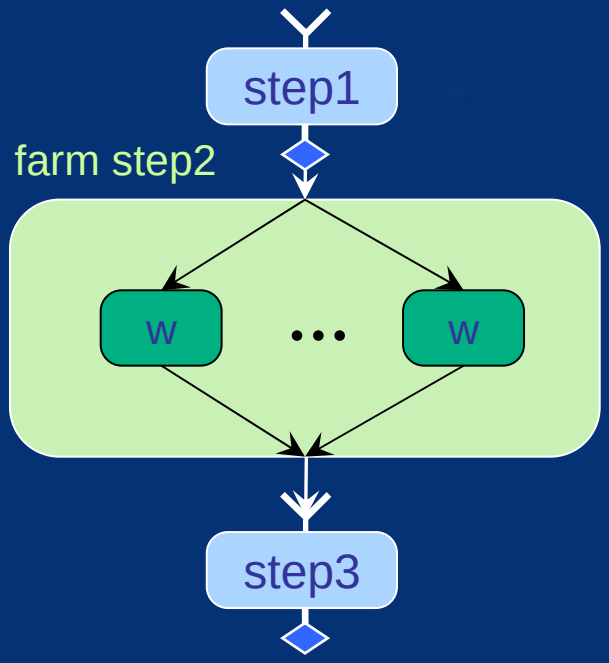


- Port cardinality

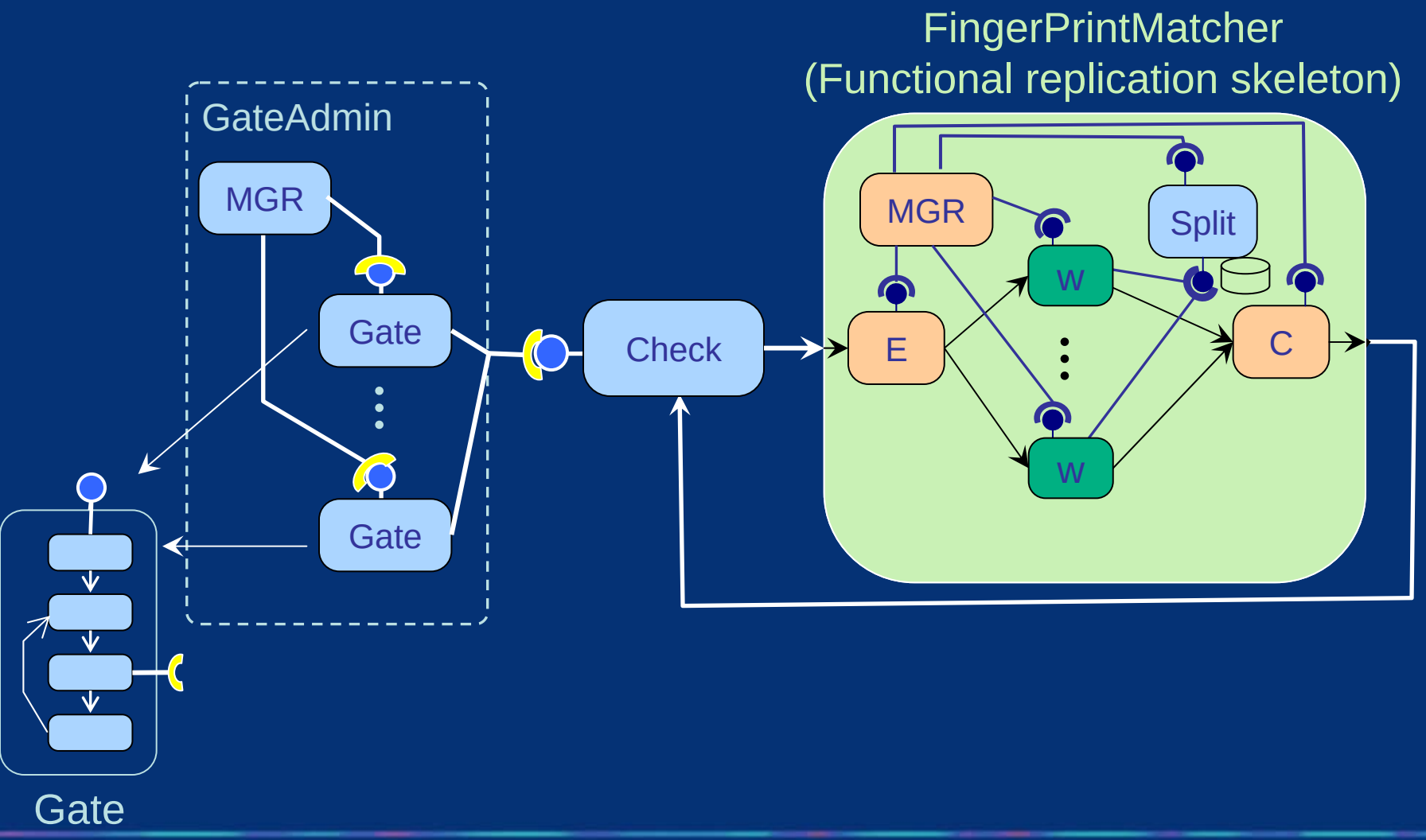


Assembly model

```
component Example{
   … Step1 and Step3 components…

   farm Step2{
      inputSkel double inS2;
      outputSkel string outS2;

      worker sequential w {
         inputSkel double inW;
         outputSkel string outW;
         component Worker{   streamIn double inW;
                             streamOut string outW;
         };
         connect outW to Worker.outW;
         connect Worker.inW to inW;
   };

instances:  Step1 step1; Step2 step2; Step3 step3;
 … Connexions step1 <=> step2 <=> step3 …
sequence ApplMain{
   exectask(step1); exectask(step2); exectask(step3);
};
```



STKM usage and benefits



Outline of the talk

- Introduction
  - Software component models
- Existing models
  - STCM: a spatio-temporal component model
  - Skeletons based models for parallel programming
- STKM: proposal of skeletons introduction in STCM
  - Objectives
  - Overview
  - Example of usage
- Conclusions and future works

Conclusions

- A combination of component models, workflows and skeletons
- Previous works
  - STCM: merging component models with workflows (Related work)
  - Skeleton models
- Contribution: STKM
  - Merging STCM with skeleton models
- All advantages in a same model

Perspectives

- Already done
  - Manual implementation prototype on top of SCA
  - Preliminary experiments results
  - (TR-0171 CoreGrid)
- Future works
  - Framework implementation for automatic generation of assemblies at execution
  - Generic skeletons constructs for easy extension with new skeletons
  - Applications implementation

Questions ?