Application-Oriented Performance Analysis in CrossGrid

Roland Wismüller

LRR-TUM, Technische Universität München

Munich, Germany

Agenda

1 **Background**

2 **Basic Approach**
- On-line performance measurement
- User-defined Metrics

3 **Research Issues**
- Semantics
- Metrics specification language
- Distributed implementation of measurements

4 **Current State**

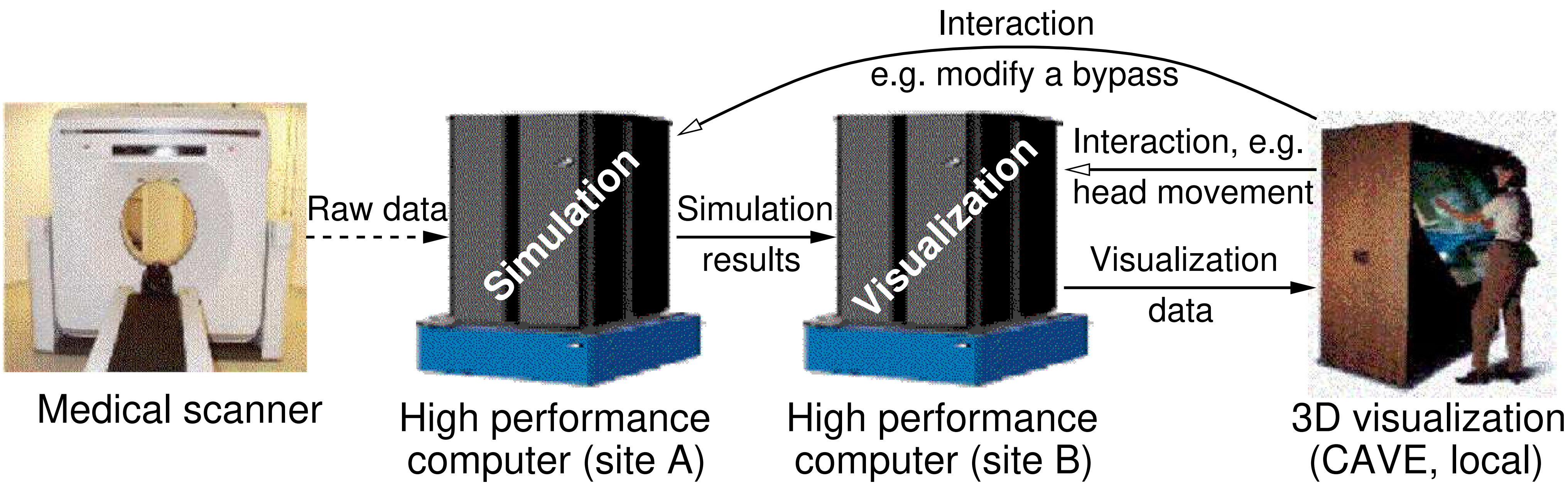
5 **Conclusions**

1 Background

EU IST-project CrossGrid

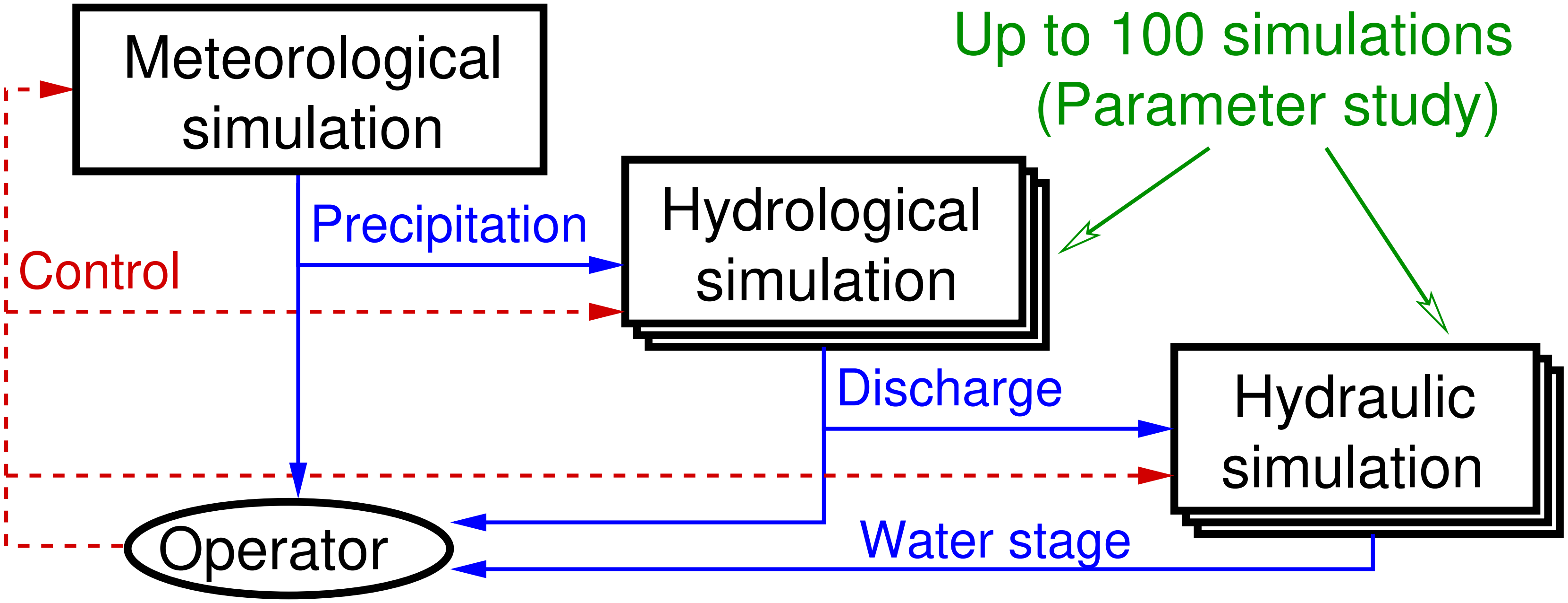
- Aim: make the Grid usable for *interactive* applications
  - four parallel / distributed applications
  - tools (test, benchmarks, performance analysis / prediction)
  - middleware (portals, scheduling, monitoring, file access)
  - testbed
- Task 2.4.1: G-PM: a practical performance analysis tool for interactive applications in the Grid
  - ICS-AGH, Cracow (Marian Bubak, Wlodzimierz Funika and others) and LRR-TUM
- Schedule: March 2002 – Feb. 2005

1.1 Medical Simulation



- Interactive visualization and simulation
  - response times are critical
- Special requirements for performance analysis
  - measurement of response time and its breakdown
  - performance data for specific interactions

1.2 Flooding crisis support



- Interactive steering
  - start / kill simulations (based on new input data / performance behaviour / available resources)
- Special requirements for performance analysis
  - convergence behavior (progress of computation)
  - performance data related to the Grid infrastructure

1.3 Aims and Requirements

- **Meaningful performance metrics in the Grid**
  - relative to the performance of the Grid infrastructure
- **Support for application specific metrics**
  - e.g.: response time, convergence rate, data for one interaction, ...
  - with minimal effort for the user
- **On-line analysis and visualisation**
  - correlation of performance behaviour and interactions
  - prerequisite for steering
- **Efficient, distributed data evaluation**
  - scalability, perturbation of application
  - avoid a centralised analysis of event traces

2 Basic Approach

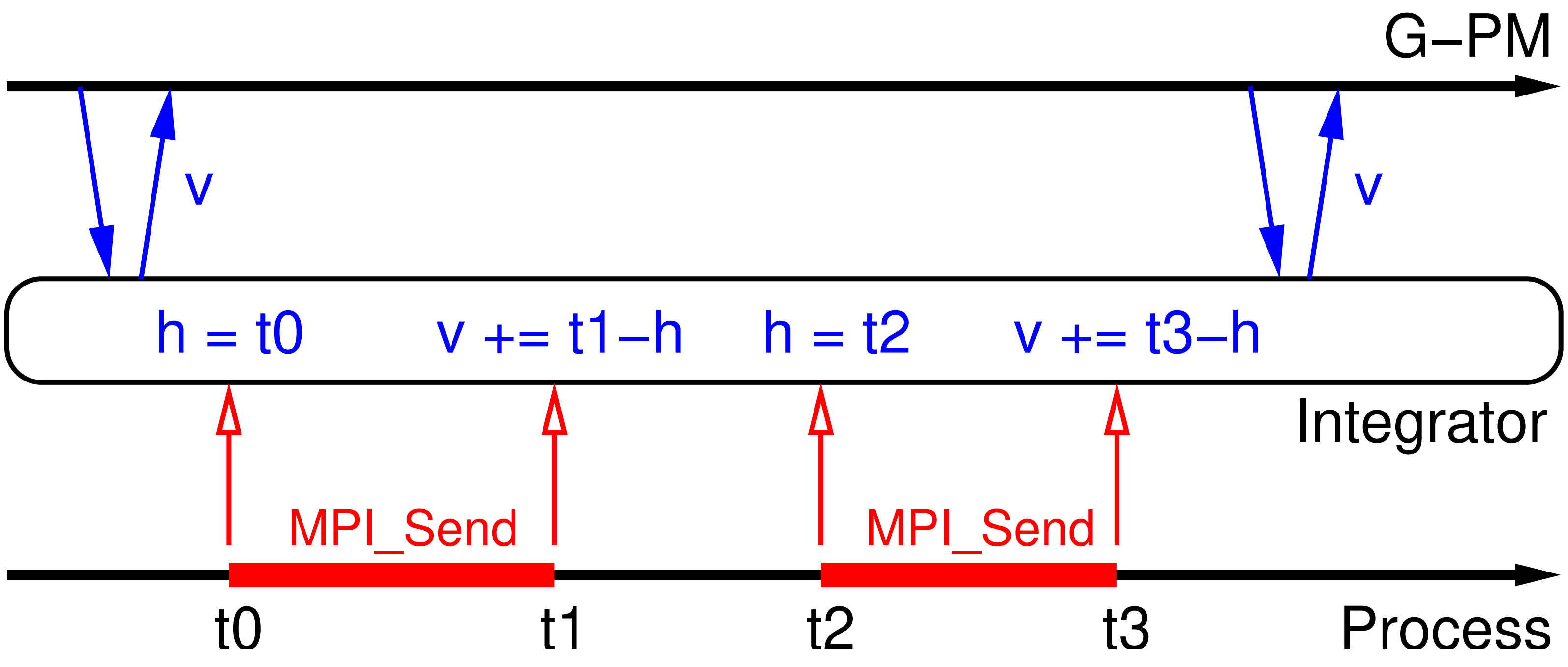
- **G-PM tool offers a set of standard metrics**
  - e.g. communication time, I/O volume; link bandwidth, ...
  - automatic instrumentation, transparent for the user
- **Application provides a minimal set of additional information**
  - relevant events inside the application
  - association between events
  - relevant data computed by the application
  - manual source code instrumentation (*probes*)
- **G-PM allows the user to define new metrics**
  - basis: existing metrics and information from probes
  - flexibility, re-use of metrics and probes

2.1 Metrics and Measurements

- **Metrics:**
  - something that can be measured (e.g. length, width, area, weight, ...)
  - independent of a specific object that is measured
  - metrics can be either measured directly or computed from other metrics (e.g. area = length * width)
- **Measurement:**
  - the actual measurement of a metrics for a specific object (or a set of objects)
  - in G-PM: measurements can be restricted based on
    - partner objects (for communication related metrics)
    - program code regions

2.2 On-line Performance Measurement

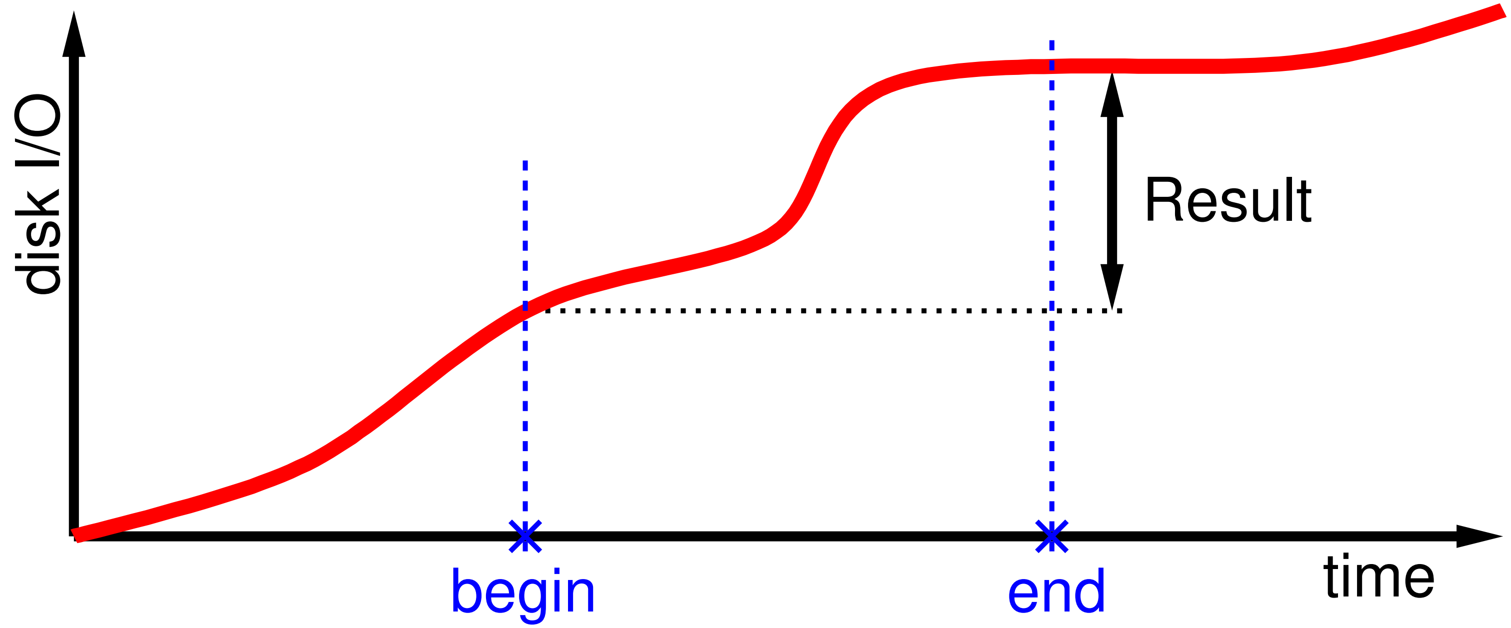
- Monitoring of *begin* and *end* events (e.g. of communication routines)
  - action: update a counter or integrator
- G-PM requests the value periodically and displays it



2.3 Principle of User-defined Metrics

Example: disk I/O volume caused by an interaction

- Basis: measurement of the total disk I/O volume
- Probes mark beginning and end of an interaction
- Result = disk I/O (*end*) – disk I/O (*begin*)



3 Research Issues

- **Performance measurement method**
  - semantics, correct implementation
- **Semantics of user-defined metrics**
  - events in distributed systems (temporal order?)
  - loss of information due to aggregation
- **Simple, user friendly specification of metrics**
- **Optimal, distributed implementation of measurements**

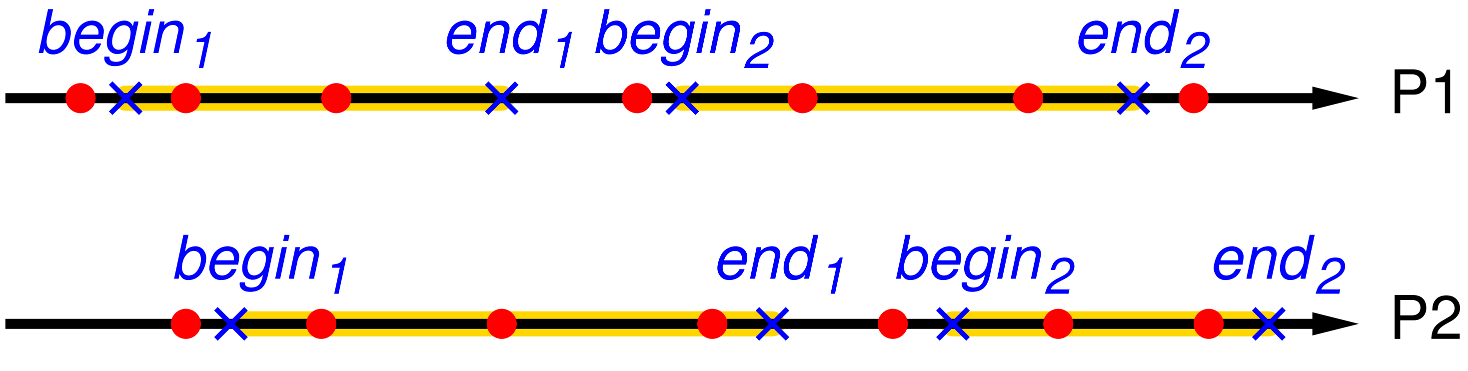
Problems of semantics and correct implementation

- **Exact definition of semantics**
  - essential, since results are basis for further computations
- **Activities with long duration**
  - how to get a measurement value between the *begin* and *end* events?
  - what if a measurement starts between a *begin* and *end* event?
- **Aggregation of results from different nodes**
  - results refer to slightly different intervals of time

3.2 Semantics of user-defined metrics

Events in distributed systems

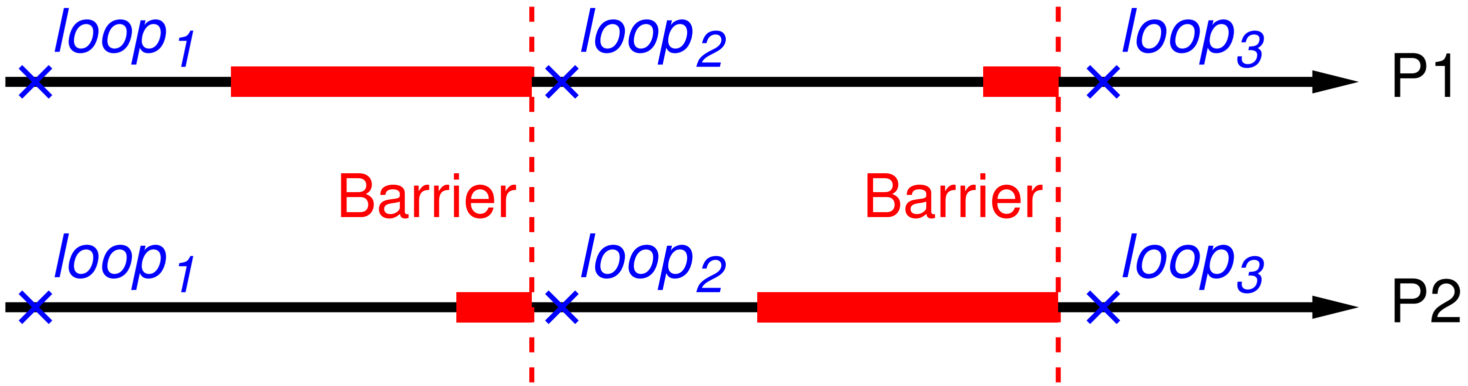
- In a distributed system, simultaneity is not well-defined
  - thus, an event cannot accurately trigger a measurement on a remote node
- Consequence: *begin* and *end* probes locally in each process
- Correspondence indicated by matching IDs passed to probes (*virtual time*)



- Drawback: more work for the user, if code is non-SPMD

Loss of information due to aggregation

- Completely abstaining from using event records can hide interesting effects
- E.g. load imbalance in a loop (waiting time at the barrier)



- Solution: record a sequence of measurement values
  - one value for each execution of the loop (*virtual time*)

3.3 Specification of metrics

- Specification must (implicitly) specify all necessary actions
  - when a measurement for this metrics is defined
  - when a probe is executed
  - when a measurement value is requested by G-PM
- Existing specification languages
  - ASL (Apart), JavaPSL (U. Vienna): performance properties
  - MDL (U. Madison): low level measurements
  - EARL (Research Center Jülich): trace analysis
- For G-PM we need a different language
  - different goals and focus
  - language is exposed to user ($\Rightarrow$ must be simple)

PMSL: a declarative metrics specification language

- Basic concepts: data flow principle, (very simple) unification

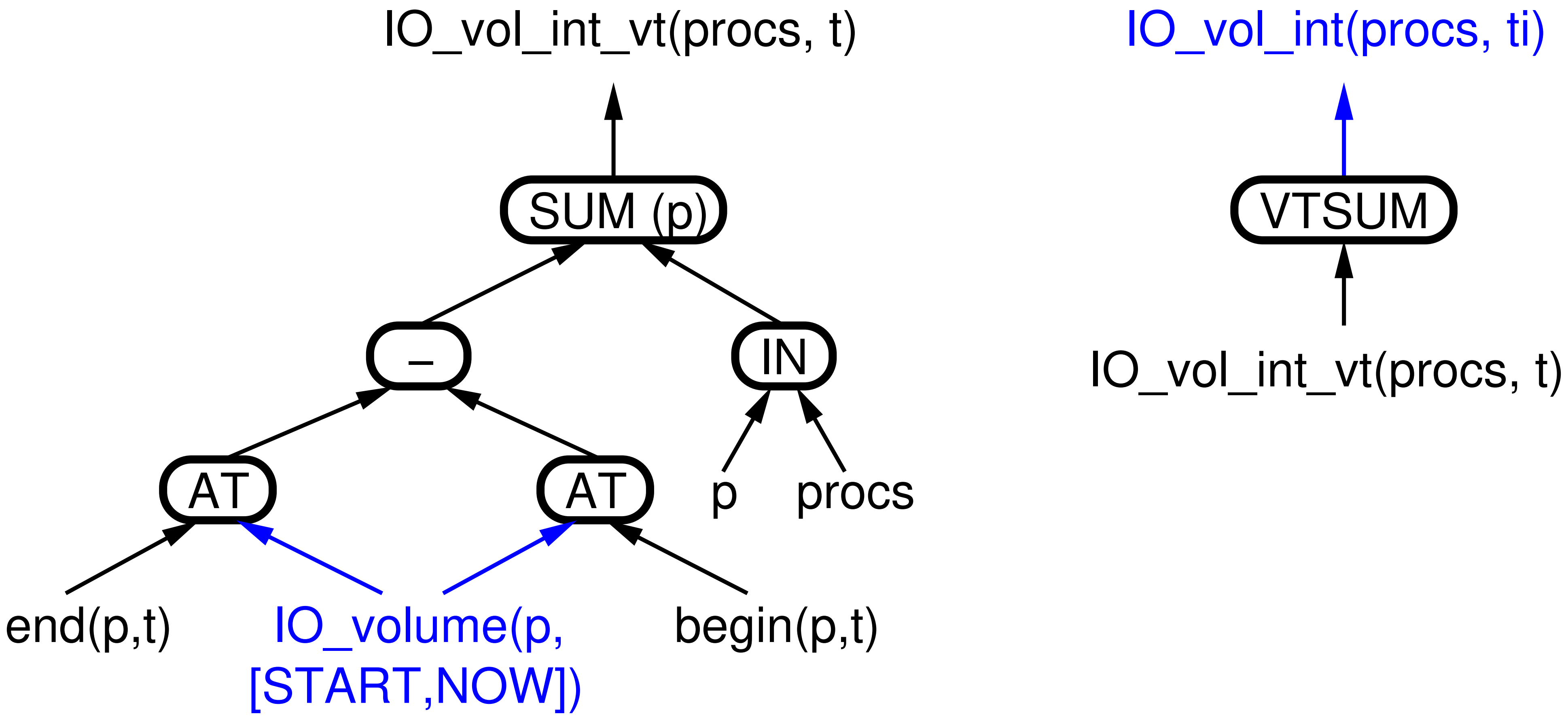
```
IO_vol_int_vt(Process[] procs, VirtualTime t)
{
  Process p; Value vol[];
  vol[p] = IO_volume(p, [START,NOW]) AT end(p, t)
         - IO_volume(p, [START,NOW]) AT begin(p, t);
  return SUM(vol[p] WHERE p IN procs);
}
IO_vol_int(Process[] procs, TimeInterval ti)
{
  VirtualTime t;
  return SUM(IO_vol_int_vt(procs, t) WHERE t.time IN ti);
}
```

3.4 Distributed Impl. of Measurements

Specification is transformed into a distributed data flow graph

- **After specification of the metrics**
  - create a graph template
  - values of parameters (object, restrictions) still unknown
- **When a measurement of this metrics is defined**
  - insert parameter values, do a partial evaluation
    - in addition: “inline” graphs for child metrics
  - transform the resulting graph
    - aggregation in time is moved down (towards event sources) as far as possible
  - distribute the graph
    - subgraphs which can be evaluated locally are sent to the distributed monitoring system

Example: graph template

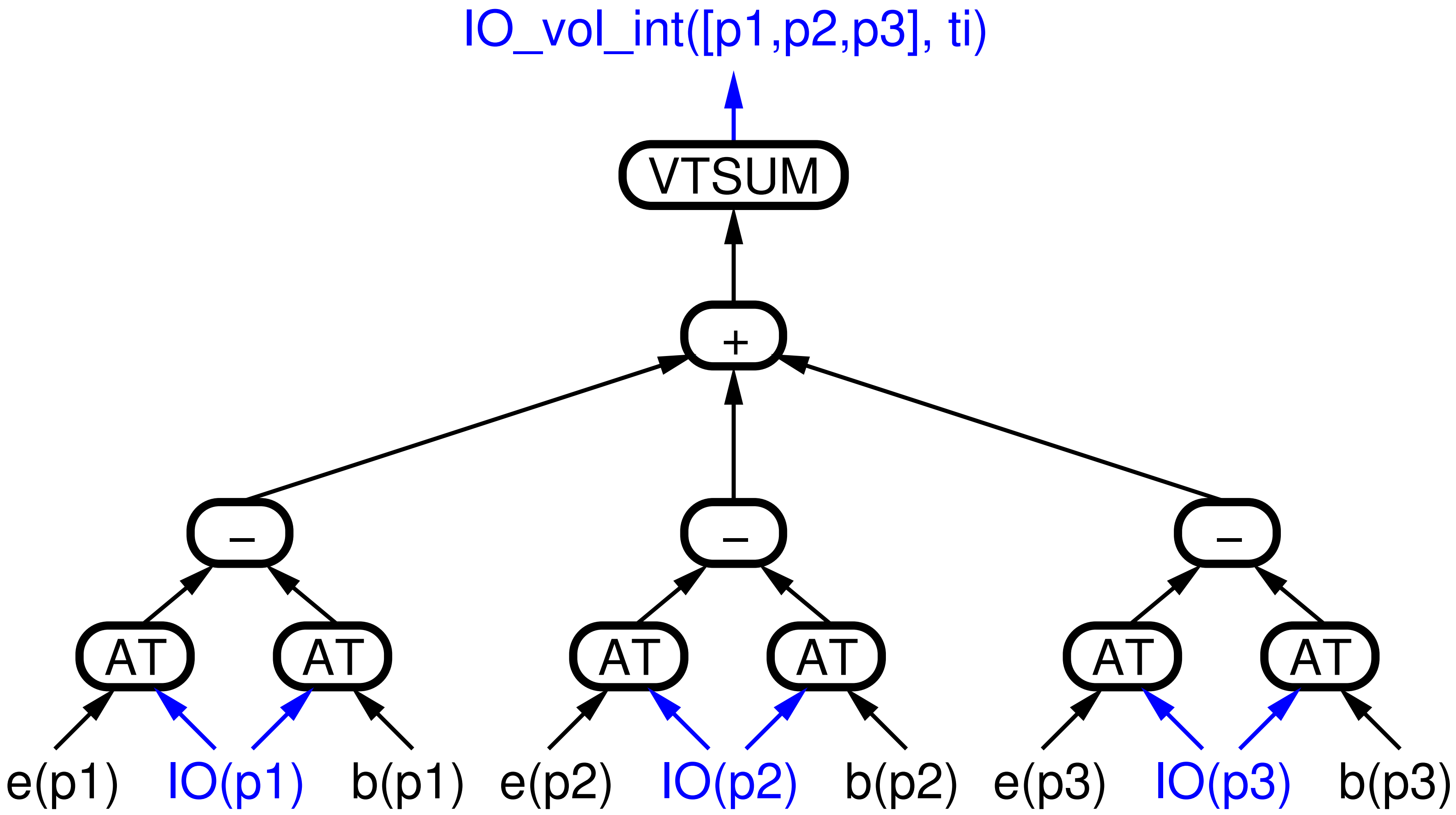


event triggered (push)

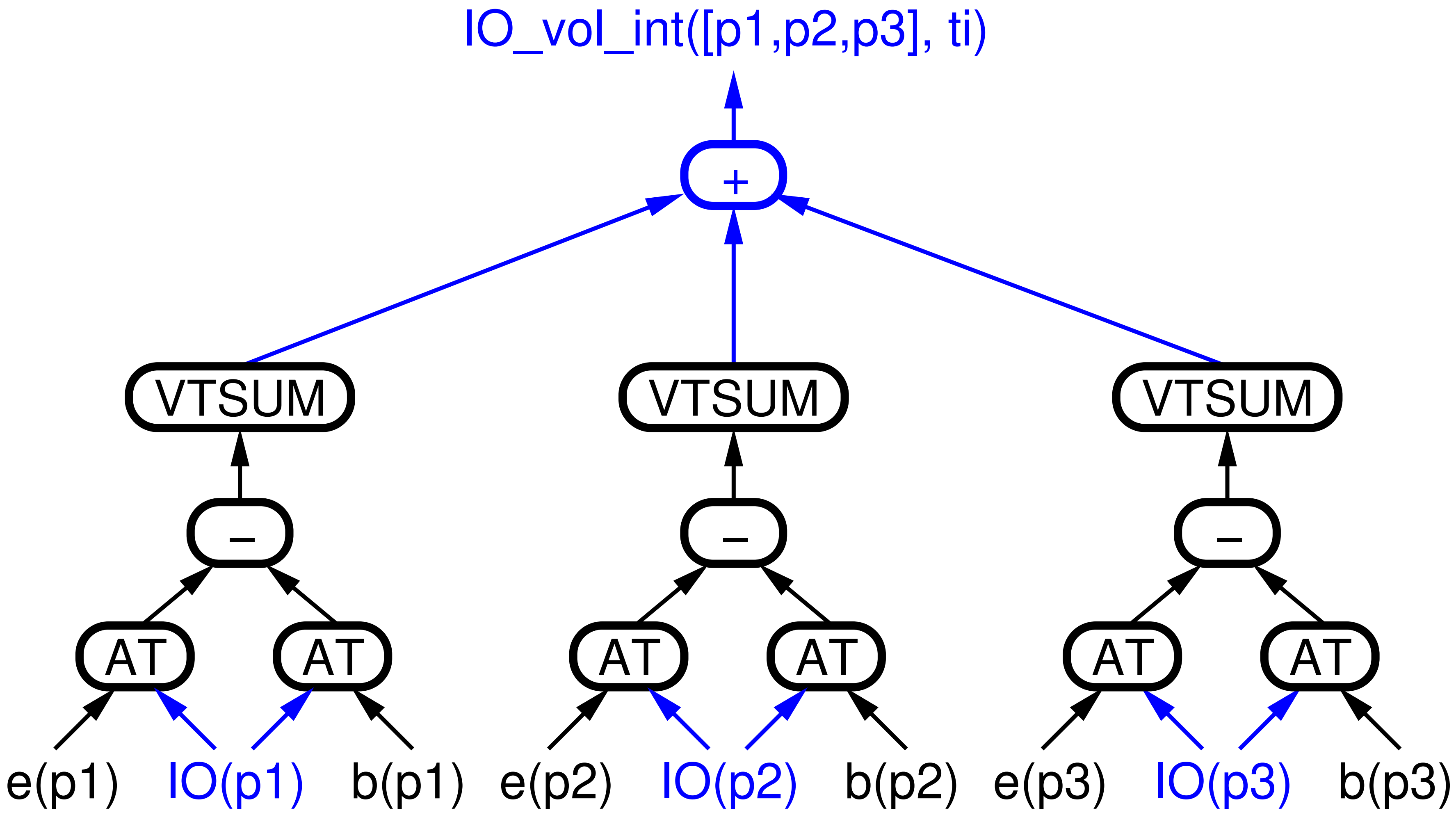
request triggered (pull)

3.4 Distributed Impl. of Measurements ...

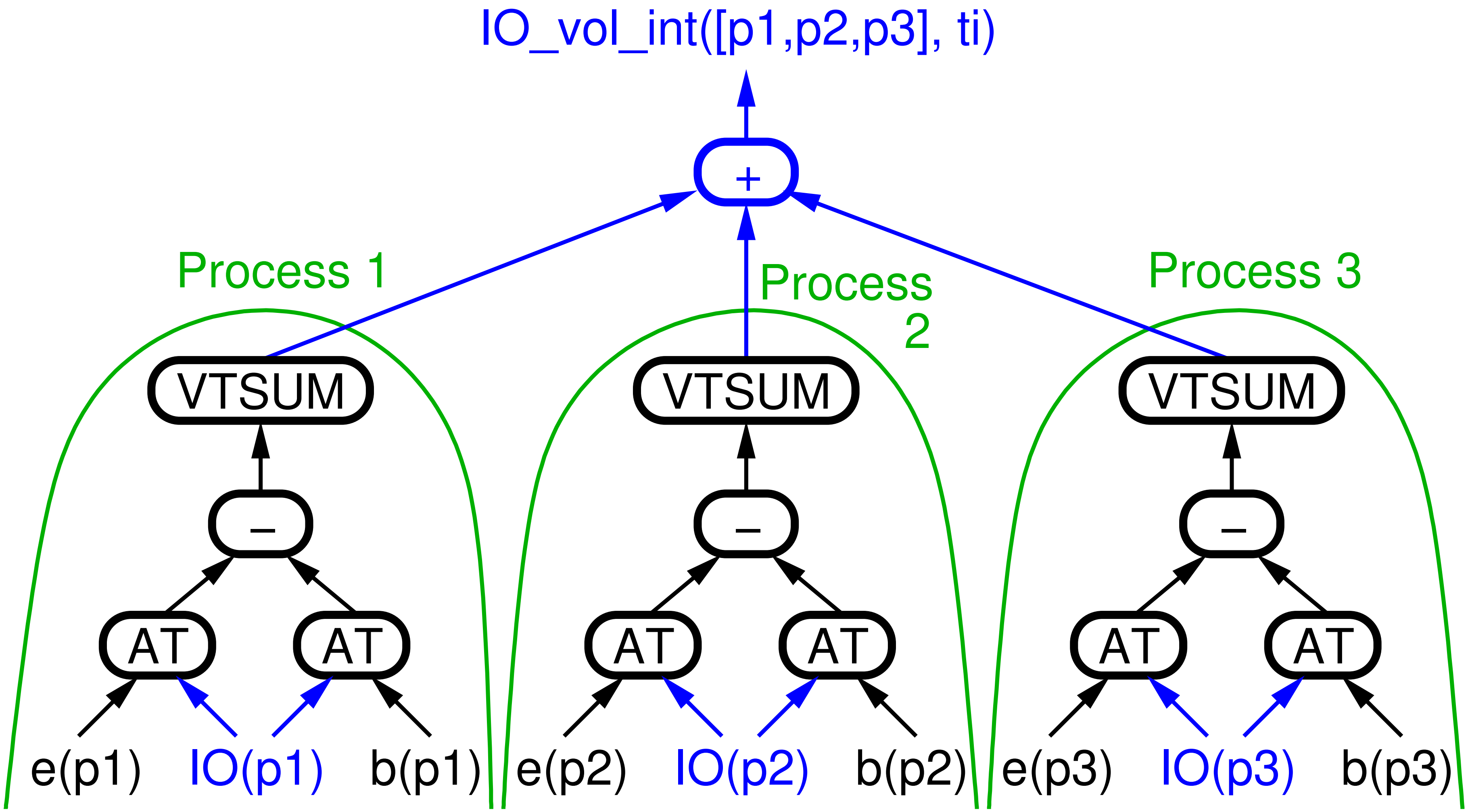
Example: graph when measurement is defined (3 processes)



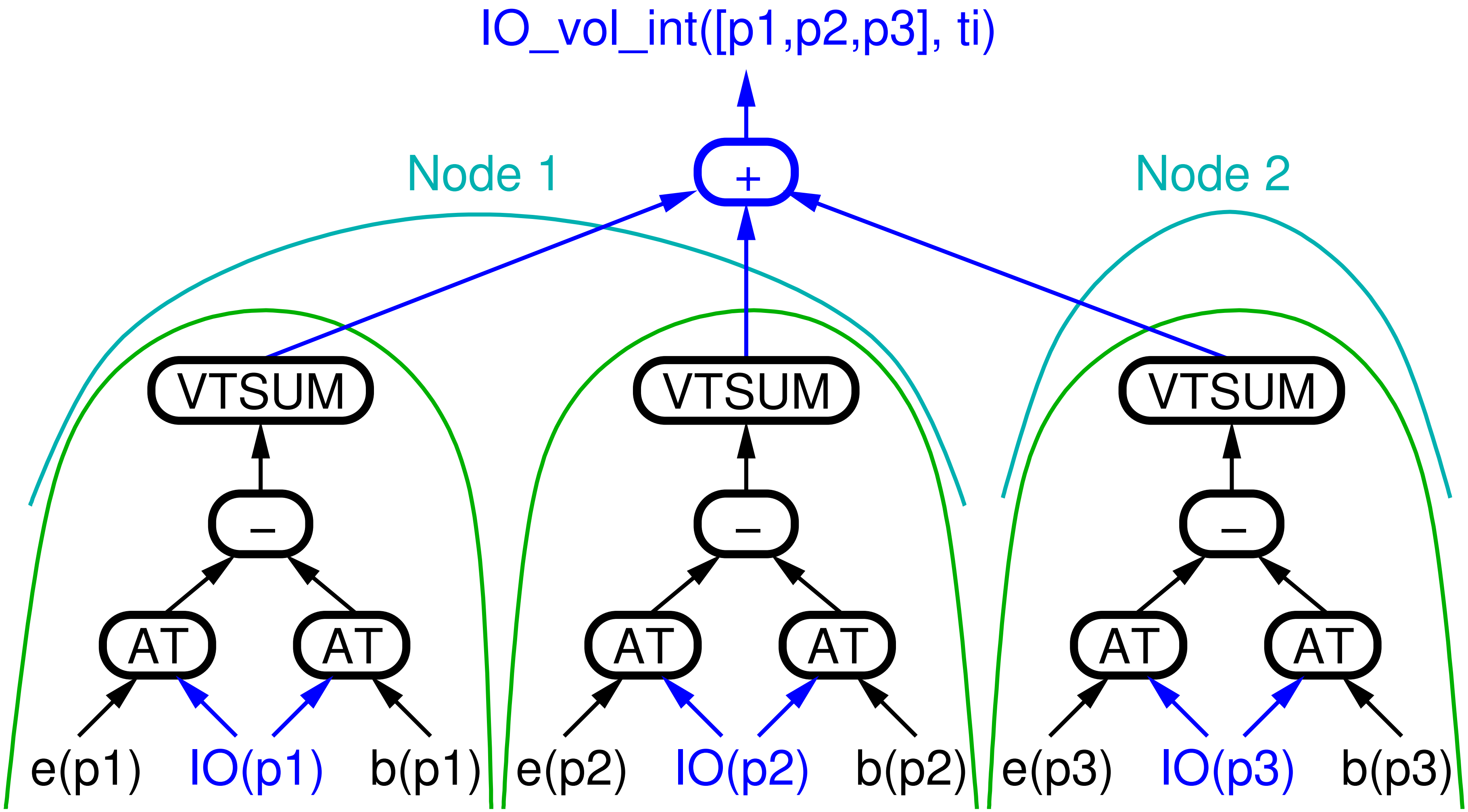
Example: interchange the sum operators



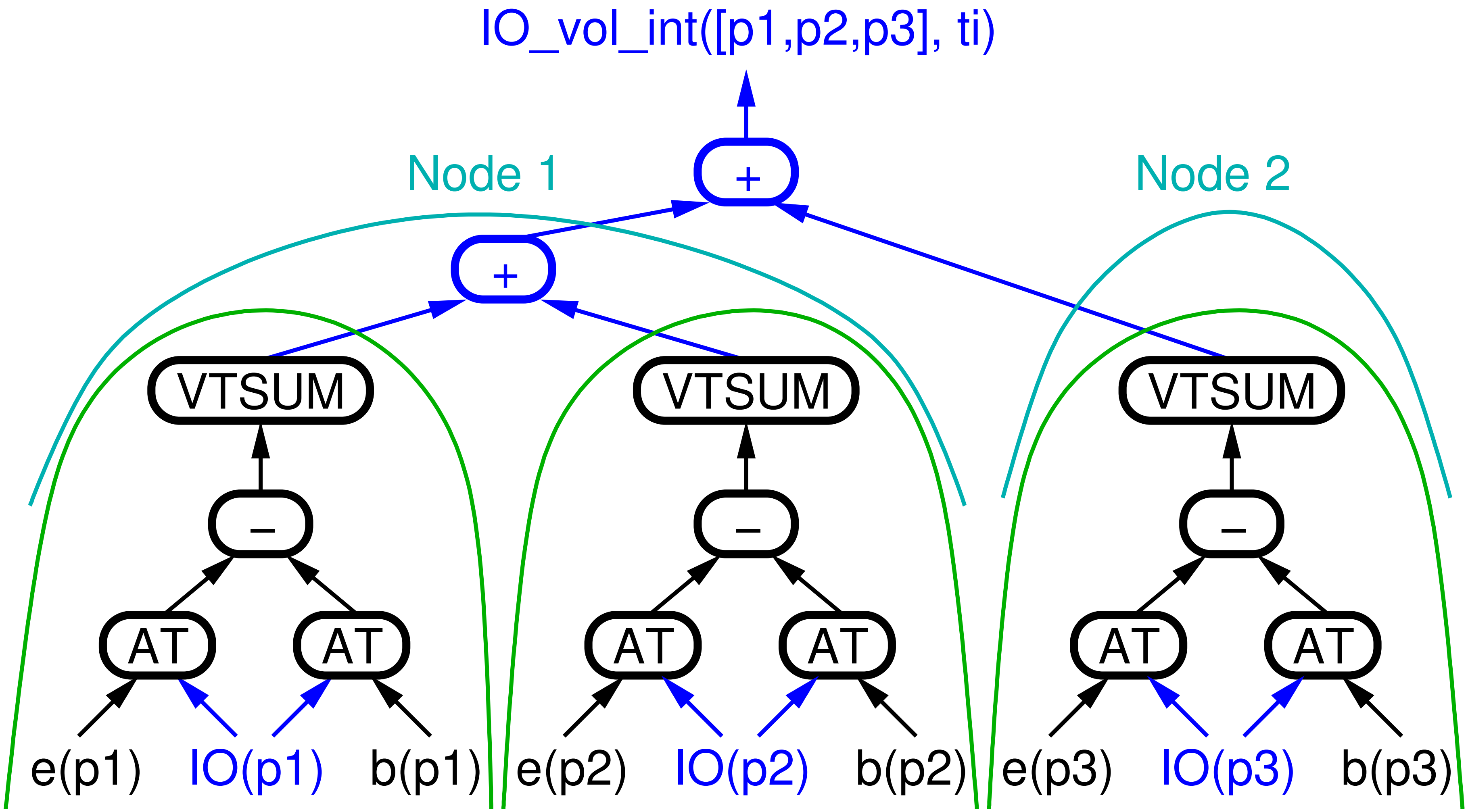
Example: locally evaluable subgraphs



Example: locally evaluable subgraphs



Example: hierarchical aggregation



3.4 Distributed Impl. of Measurements ...

Infrastructure for the evaluation of the graphs

- Monitoring system OCM-G, monitoring interface OMIS

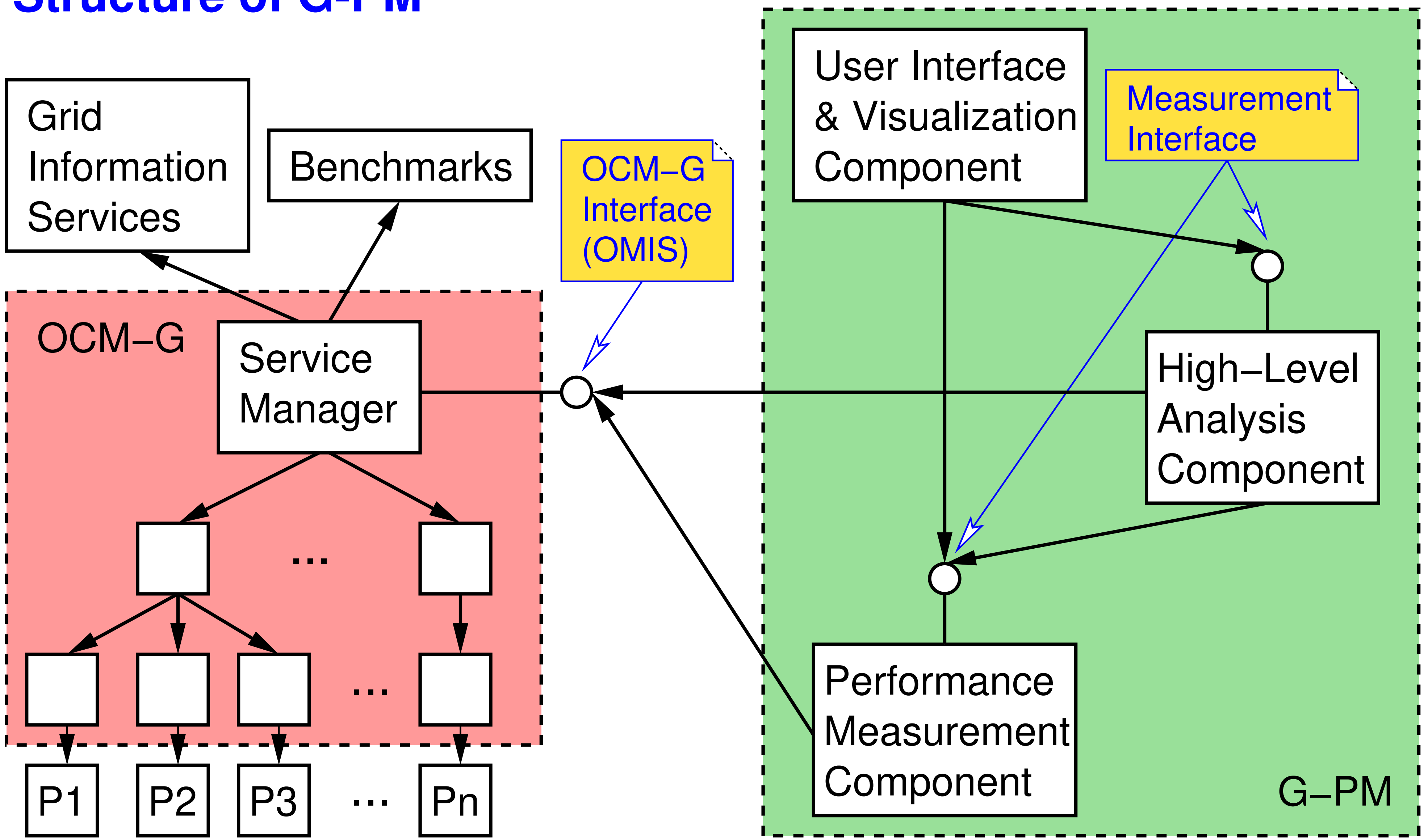
Concepts of OMIS/OCM-G

- Object based model of the target system
- Programmable relations between events and actions

```
thread_executes_probe([p_1,p_2], "begin") :
    pa_integrator_read(pa_i_1, 0)
    print([$proc])
```

- Object sets, location transparency
- Extensibility

Structure of G-PM



4 Current State

- First version of G-PM is available
  - communication volume and time, CPU and memory usage
  - bar graph and curve diagrams
  - demonstrator for user-defined metrics
- First version of OCM-G for use in the Grid
- Definition of PMSL completed
- **Current Work:**
  - implementation of a parser for PMSL
  - creation and execution of data flow graphs
- **Future Work:**
  - optimization of data flow graphs

5 Conclusions

- Interactive Grid applications demand
  - on-line performance analysis
  - application specific metrics
    - provide set of standard metrics
    - allow for application specific events and data
    - provide specification language to define new metrics
  - optimized, distributed implementation of measurements
    - requires a flexible infrastructure for application monitoring

- Still many research issues need to be addressed
  - exact semantics of measured values!