Build Common Performance Measurer using Java Web Dynpro

Jenny Zhang, Developer xRPM

April 2005

THE BEST-RUN BUSINESSES RUN SAP

Agenda

POWERED BY SAP NetWeaver™

- Measurer for performance tuning
- Goals of performance measurer and KPI
- Architecture and design
- Detailed usage
- Demo

THE BEST-RUN BUSINESSES RUN SAP

Measurer for performance tuning

- **Performance measurer is for the purpose of measuring system performance and provide the baseline for performance tuning.**
- **Why need performance tuning**
  - **It is for system quality: relief pain of user.**
    - **People expect things happen within 3 seconds.**
  - **System runs slow simply because it has not been performance tuned.**
  - **It really improves product performance.**
  - **Performance tuning is one of the most important things for a product instead of it is only cool.**

Experience

Before doing performance tuning project:

- **BISDK's performance was becoming a big concern although we used a lot of cutting edge technologies: JCA, JMI, ODBO, CWM, MMR etc.**
  - **System connection time is more than 10 seconds.**
  - **Getting SAP variables is more than 45 seconds.**
  - **Query execution is not linear.**
  - **…**

After doing performance tuning project:

- **System bottlenecks have been identified and removed, system performance has dramatically improved in certain area.**
  - **System connection time is less than 3 seconds by improving locale info impl.**
  - **Getting SAP variables is less than 1 second by reducing system roundtrip.**
  - **Makes query execution linear by eliminating empty cells.**
  - **…**

Agenda

- Measurer for performance tuning
- Goals of performance measurer and KPI
- Architecture and design
- Detailed usage
- Demo

Performance Measurer: Goal and main performance KPI

Goal

- A reusable Web Dynpro component for measuring performance.
  - Measure result views are able to be embedded in other webdynpro view.
- Being able to track backend call in xRPM 4.0.
- Being able to track the overall product performance, identify areas of performance bottlenecks, provide ways for performance analysis.
- Provide performance baseline to evaluate the performance of xRPM 4.0 from front end.

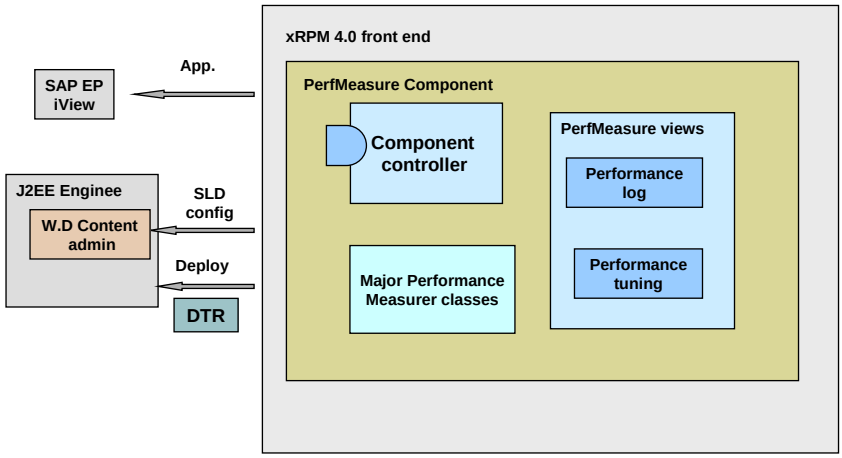
Main Performance KPI

- CPU Time: How long it takes in order to finish specified operation in milliseconds.
- System round trip: How many times you need to call back to backend.
- Native Time: How long it takes to call back to backend in order to finish specified operation in milliseconds.

Agenda

- Measurer for performance tuning
- Goals of performance measurer and KPI
- Architecture and design
- Detailed usage
- Demo

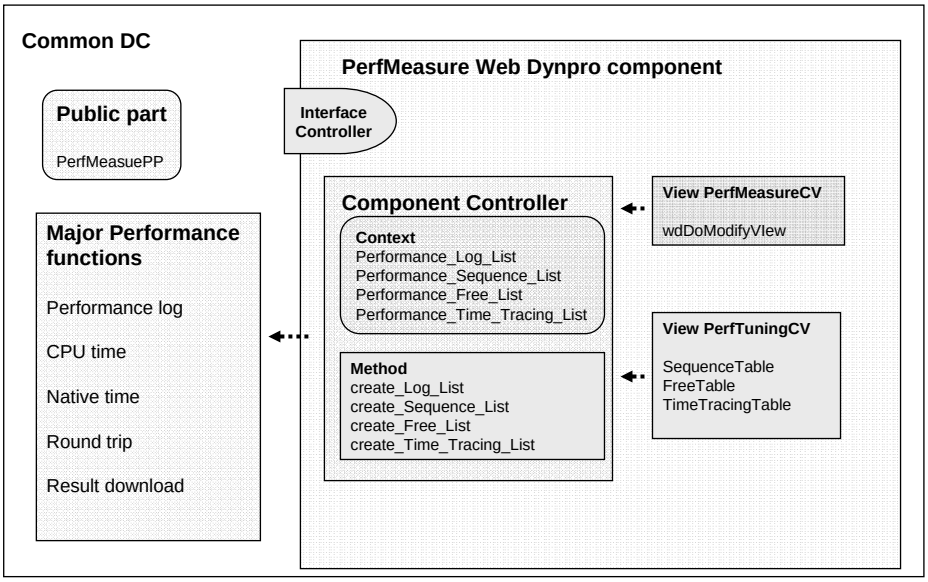
Overall architecture under xRPM4.0



Performance Measurer Design

- **Reusable web dynpro component**
- **Performance functions: main performance functions for measuring CPU time, Native time, and System round trip.**
- **Component controller: Handling performance measure data and access to performance functions .**
- **Controller context: performance measurer's data.**
- **View/Controller: View for performance log. Tables to render performance data, functions for view.**
- **Download result: download results to client side.**

Design diagram



Agenda

- Measurer for performance tuning
- Goals of performance measurer and KPI
- Architecture and design
- Detailed usage
- Demo

Performance Log

- A reusable view that can be embedded in other web dynpro view container.
- It can show backend call CPU time, backend call input and output XML.
- RFC model execution:

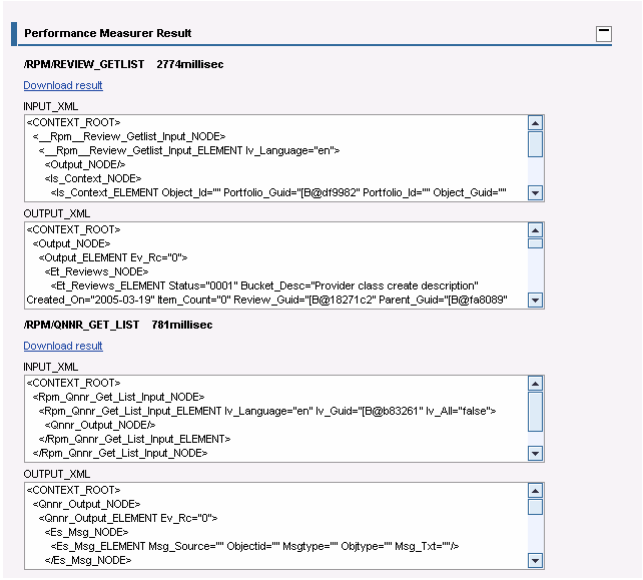
Using:

```
perfMeasure.LogExecute(inputNode, outputNode);
```

Instead of:

```
InputNode.currentInputElement.modelObject().execute();
```

Performance Log screen shot



Ways of measuring - Measuring in sequence

- **Measuring system performance in sequence.**
- **User starts a measure, followed by ending the measure in sequence without nesting with other measure.**
- **Advantage: User does not need to provide unique description keys for start and end.**

```
perMeasure.startSequence(description);
...
perfMeasure.endSequence();
```

Ways of measuring - Free style

- Measuring system performance on demand: either in sequence or nested.
- User starts a measure with a description key and end a measure with the same description key.
- The measure description key for one measuring item must be unique.
- Advantage: User could do nested measurement.

```
perfMeasure.startFree(description1);
...
perfMeasure.startFree(description2);

perfMeasure.endFree(description1);
...
perfMeasure.endFree(description2);
```


Ways of measuring – Time Tracing

- Simply measuring CPU time from one point to another.
- User add in time tracing point at the location where the user wishes to get time.
- Advantage: Flexible to get a time tracing report from one point to another.
- Simple yet efficient.

```
perfMeasure.addTimeTracing(description);
```

Measuring native time and system round trip

- **User adds native function execution time right after backend call.**
- **User increases the counter of system round trip right after backend call.**
- **Advantage: Know exactly how many roundtrips causes how long time within each CPU time measurement.**

```
long timer = System.currentTimeMillis();
<Native call>;
perfMeasure.addNativeTime(System.currentTimeMillis()-timer);
perfMeasure.addRoundTrip();
```


Measuring Result

Performance Tuning Result

Download all result

Measuring performance in sequence

	Description	CPUTime	RoundTrip	NativeTime
☐	RpmModelCust.sequence1	10	0	0
☐	TestPerfMeasureCV.sequence1	3,695	2	3,695
☐				
☐				
☐				

1 of 2

Measuring performance in free style

	Description	CPUTime	RoundTrip	NativeTime
■	RpmModelCust.free1	10	0	0
■	TestPerfMeasureCV.free1	2,894	1	2,894
■	TestPerfMeasureCV.free2	3,695	2	3,695
☐				
☐				

1 of 3

Measuring performance using time tracing

	StartLocation	EndLocation	CPUTime
☐	RpmModelCust.time1	RpmModelCust.time2	10
☐	RpmModelCust.time2	PerfMeasure.time1	10
☐	PerfMeasure.time1	TestPerfMeasureCV.time2	2894
☐	TestPerfMeasureCV.time2	TestPerfMeasureCV.time3	801
☐			

1 of 4

Agenda

- Measurer for performance tuning
- Goals of performance measurer and KPI
- Architecture and design
- Detailed usage
- Demo

Copyright 2005 SAP AG. All Rights Reserved