



IBM Research

# Advanced Computing Technology Center Update

I-Hsin Chung  
IBM T.J. Watson Research Center  
[ihchung@us.ibm.com](mailto:ihchung@us.ibm.com)

# ACTC Toolkit Activities

- **HPC Toolkit**

- Traditional framework for interactive application performance analysis
- **Analysis-based** as opposed to tuning-based

- **HPCS Toolkit (DARPA Productivity Program)**

- Next generation framework for **automated intelligent-assist** of application performance tuning
- Emphasis on **Tuning**
- “Automated” .....not automatic.

# IBM HPC Toolkit – Feature Highlights

- **One unified package (AIX, Linux/Power)**
  - One common visualization GUI
  - Simultaneous instrumentation for all tools (one run!)
  - Integrated tools for MPI, OMP, processor, memory etc
- **No alteration of source code required**
- **Interactive (GUI) or Programmable (Scriptable) interface**
- **Dynamically activate/deactivate data collection and change what information to collect**

## IBM HPC Toolkit – New in 2007

- **Dynamic CPU profiling**
- **Redesigned MPI Profiler/Tracer component**
  - Dynamic MPI profiling capability
  - Support for MPI-IO in MPI profiling and tracing
- **Redesigned hardware counter library**
- **I/O Performance Profiling**
- **New “IDE” features in GUI**
  - binary instrumenter
  - File editing
  - Visual source line traceback
  - Job launcher
- **Support for more platforms - BG/P, 64 bit Linux**

# Scalability Improvement – CPU profiling

- **Based on binary rewriting**
- **Simulate gprof**
  - Intercept function calls
    - injects stack walking functions to gather call graph information
    - Use hash tables to record call graph info
  - Splice in the sampling function
    - At the start of the user code execution
  - Splice in the output generation function

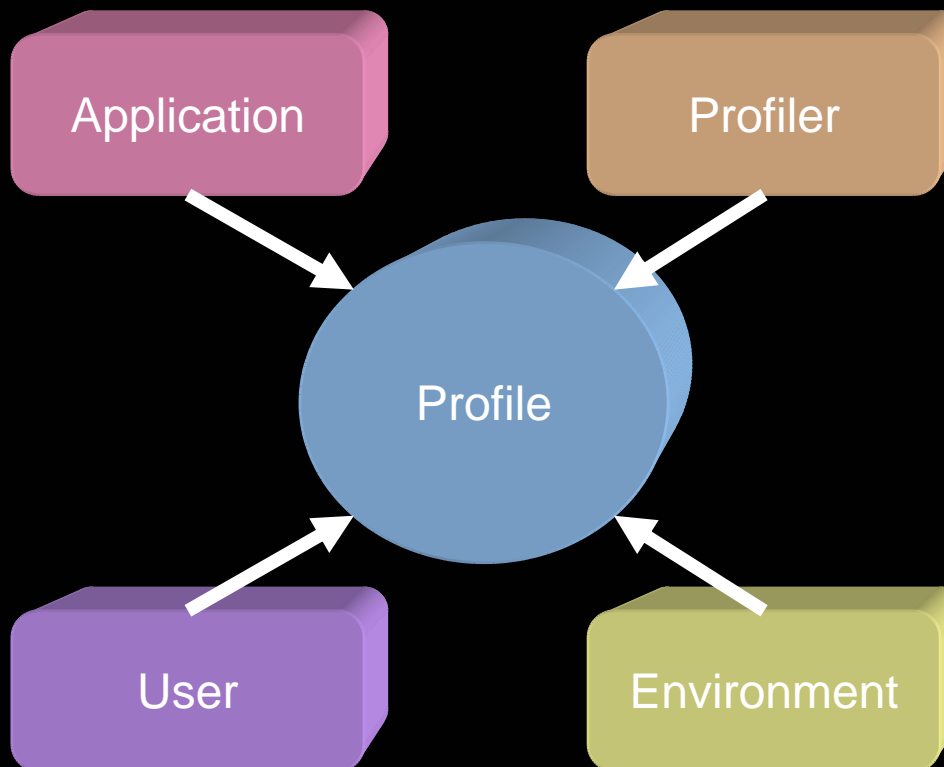
## Advantages of Improved CPU Profiler

- **Able to profile functions in precompiled libraries**
- **Avoid many of the dangling function calls in the call tree**
- **Especially useful for math libraries and MPI libraries.**

## Scalability Improvement – Configurable MPI Profiling

- **Software specific info.**

- Code segment



- **“Tool factor”**

- Memory usage

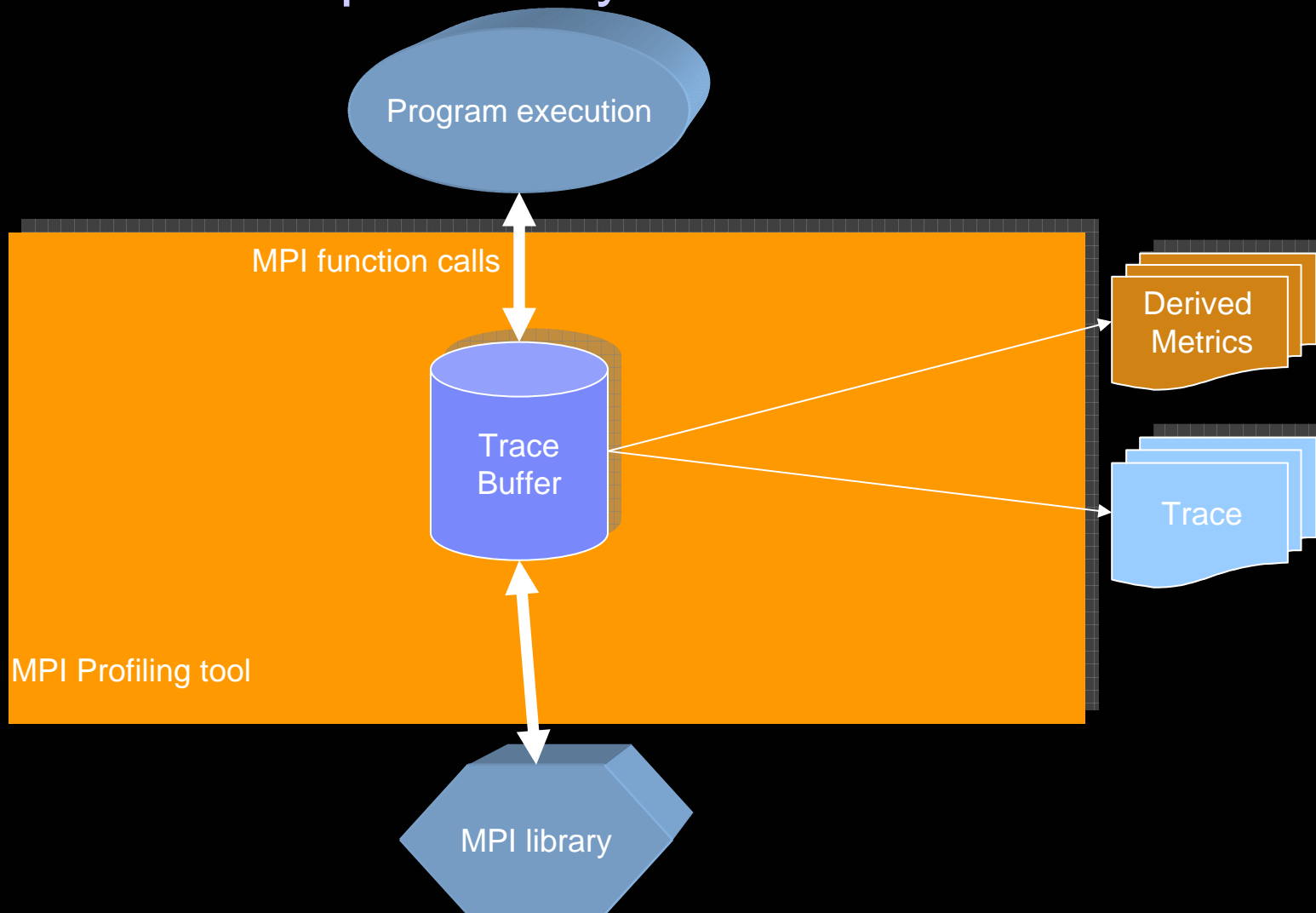
- **User preference**

- User-defined metrics

- **System info.**

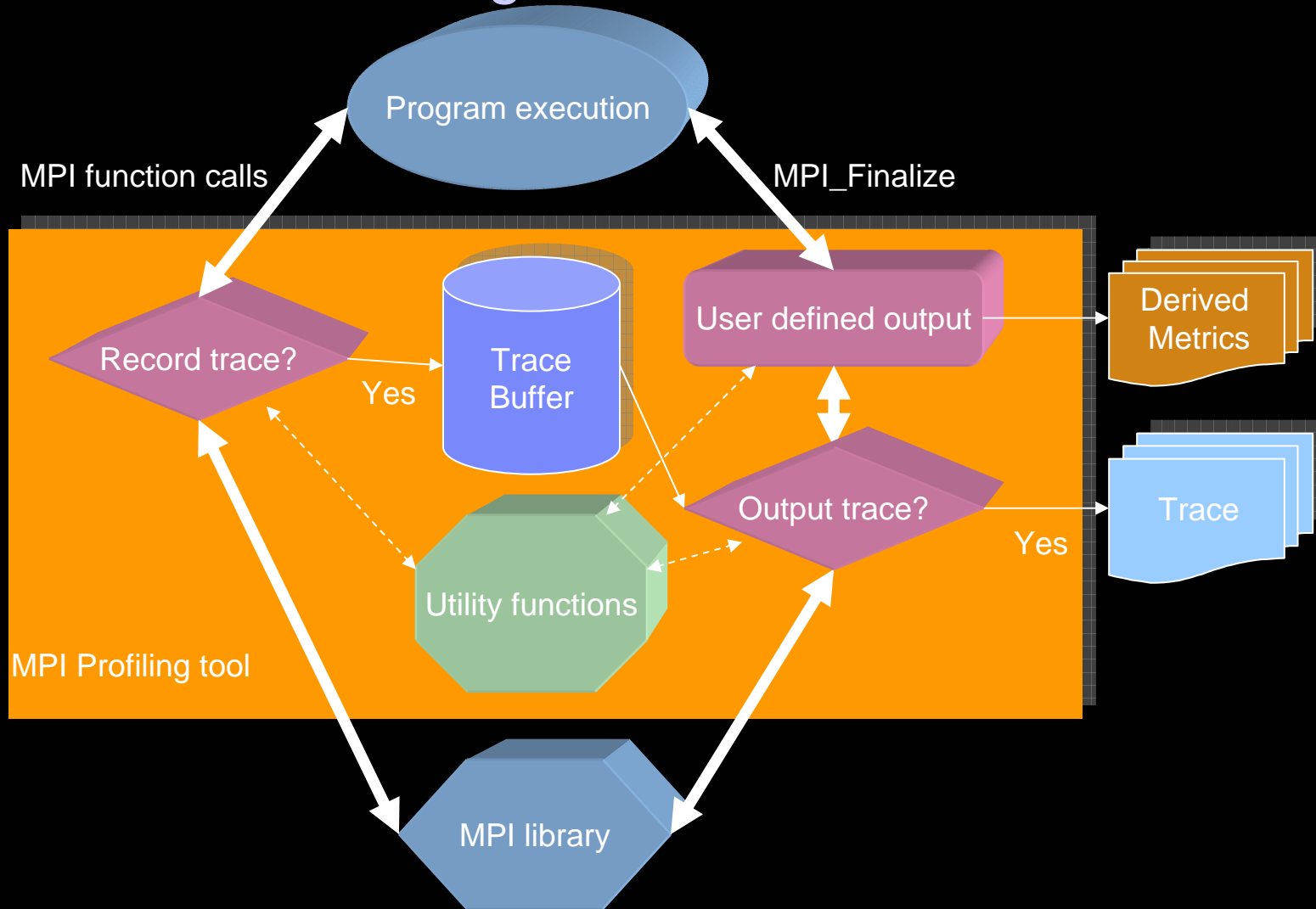
- Node location

# Implementation - previously





# Implementation - Configurable



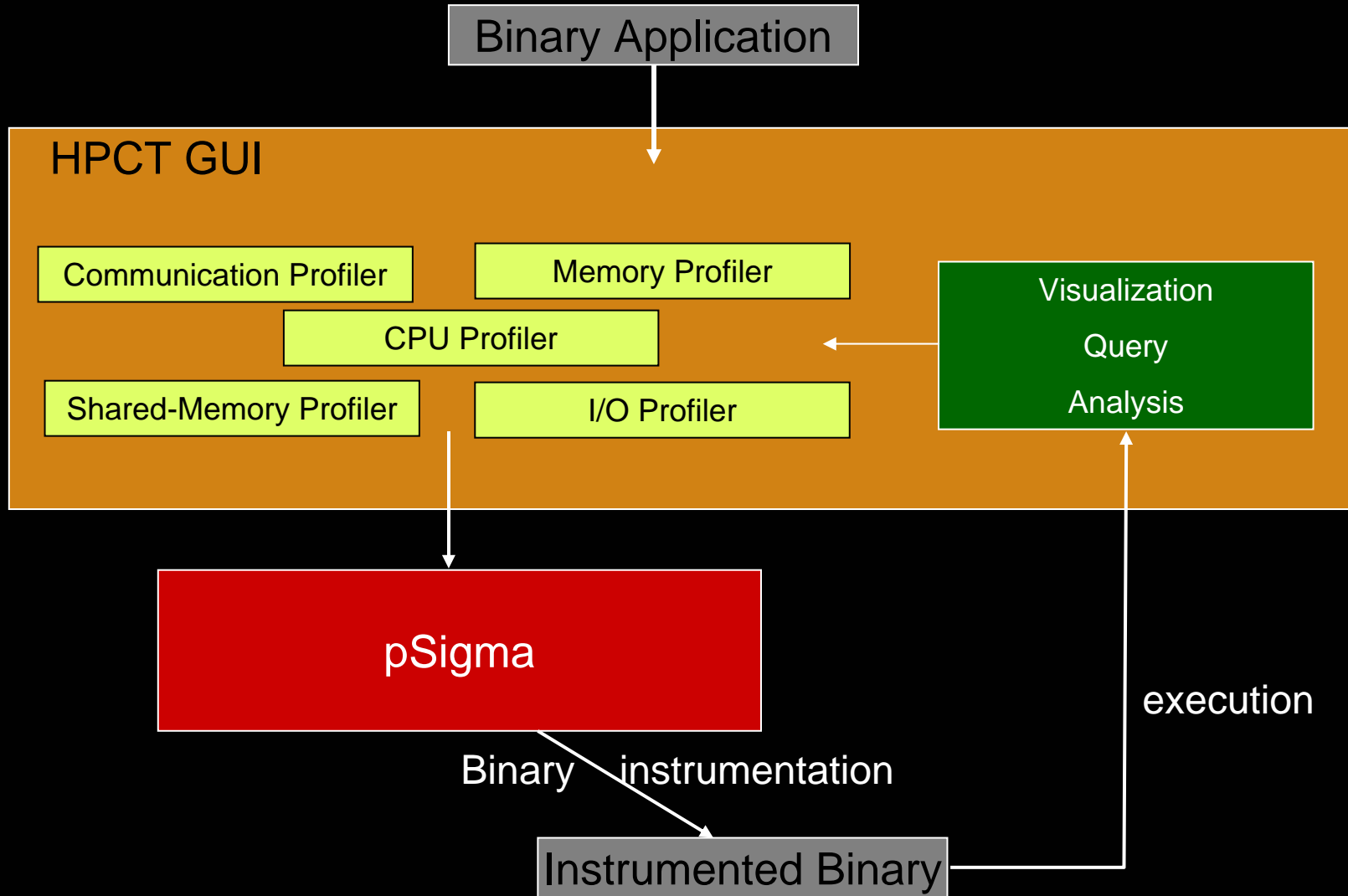
## Update – Hardware Counter Monitoring

- **Generic executable/library**
  - One executable fits all (for Power[4,5,5+,6]/AIX)
  - Support more platforms
- **Nested loop no longer necessary**
  - Support partially overlapped instrumented code segment
- **Linux/Power**
  - Replace underlying perfctr layer with perfmon2

## Update – Instrumentation Framework (pSigma)

- **IBM HPC is integrated into the pSigma infrastructure**
  - Symbolic/Binary instrumentation engine
- **Modularization**
  - Binary Analysis System
  - Binary Instrumentation System
- **Availability**
  - Supports 32/64 bit application on Linux/Power
  - Supports 32 bit application on AIX/Power

# Structure of the IBM HPC toolkit



# HPCT GUI: Graphical Instrumentation, Visualization and Analysis

**Instrumentation**

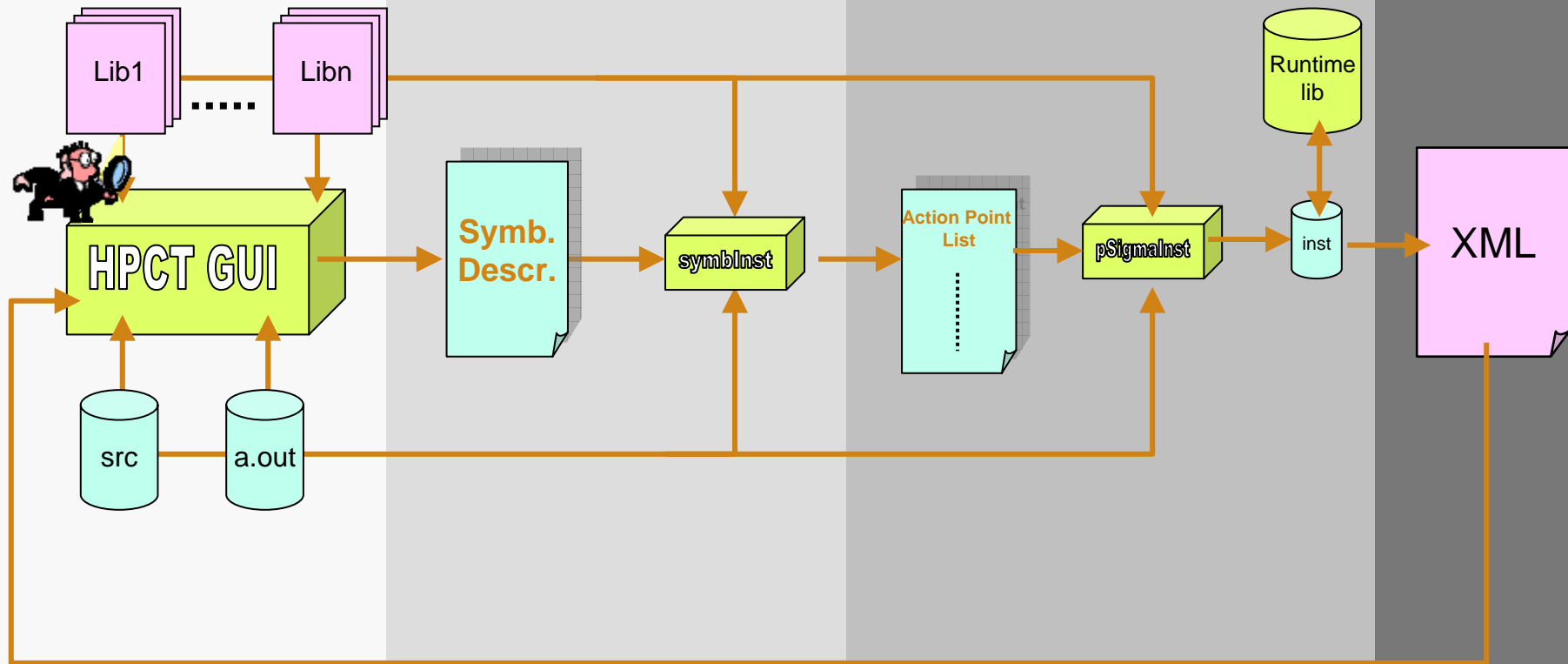
**Visualization**

**Analysis**

**Symbolic Binary Instrumentation**

**Action Point Binary Instrumentation**

**Visualization**



IBM ACTC PeekPerf: Main Window

File Tools Options Action

HPM | MPI | SIGMA | DPOMP

HPM_EVENT_SET	60
HPM_DIV_WEIGHT	5
HPM_NUM_INST_PTS	100
HPM_WITH_MEASUREMENT_ERROR	1
HPM_MEM_LATENCY	400
HPM_L3_LATENCY	102
HPM_L35_LATENCY	150
HPM_L2_LATENCY	12
HPM_L25_LATENCY	72
HPM_L275_LATENCY	108
HPM_TLB_LATENCY	700

swim\_omp

Label	Count	ExcSec	IncSec
-ALL	1	0.006	1.062
-Calc1	102	0.006	0.263
-Calc2	102	0.072	0.373
-Calc3	100	0.098	0.379
-Initial	1	0.042	0.042
-Loop 100	102	0.119	0.119
-Loop 200	102	0.178	0.178
-Loop 300	100	0.209	0.209
-MPI Calc1 end	102	0.08	0.08
-MPI Calc1 start	102	0.05	0.05
-MPI Calc2 end	102	0.068	0.068
-MPI Calc2 start	102	0.055	0.055
-MPI in Calc3	100	0.072	0.072
-loop 110	102	0.009	0.009

calc1.f | calc2.f | calc3.f | swim\_omp.f | swim\_omp.f | calc1.f | calc2.f | calc3.f

```

70         if(taskid.eq.MASTER)then
71             timei =rtc()
72             WRITE(6,*) 'SPEC benchmark 102.swim'
73
74             WRITE(6,*) ' '
75             endif
76         C
77         C Initializing array req with MPI_REQUEST_NULL
78         C
79             do i=1,16
80                 req(i) = MPI_REQUEST_NULL
81             end do
82
83         C           REQUEST PROCESSORS FOR MICROTASKING
84         C SPEC removed CMIC$ GETCPUS 4
85         C
86         C
87         C           INITIALIZE CONSTANTS AND ARRAYS
88         C
89         hpmInit (1)
90         hpmStart (5)
91         CALL INITAL
92         hpmStop (5)
93
94         if(taskid.eq.MASTER)then
95             WRITE(6,390) N,M,DX,DY,DT,ALPHA,ITMAX
96             390 FORMAT(' NUMBER OF POINTS IN THE X DIRECTION',
97                 1 ' NUMBER OF POINTS IN THE Y DIRECTION',
98                 2 ' GRID SPACING IN THE X DIRECTION',
99                 3 ' GRID SPACING IN THE Y DIRECTION',
100                4 ' TIME STEP',
101                5 ' TIME FILTER PARAMETER',
102                6 ' NUMBER OF ITERATIONS',
103                )
104             MNMIN = MIN0(M,N)
105             C initial data writes removed for SPEC
106             C WRITE(6,391) (POLD(I,I),I=1,MNMIN)
107             C 391 FORMAT('/' INITIAL DIAGONAL ELEMENTS OF P ', //
108             C WRITE(6,392) (UOLD(I,I),I=1,MNMIN)
109             C 392 FORMAT('/' INITIAL DIAGONAL ELEMENTS OF U ', //
110             C WRITE(6,393) (VOLD(I,I),I=1,MNMIN)
111             C 393 FORMAT('/' INITIAL DIAGONAL ELEMENTS OF V ', //
112             C DETERMINE OVERHEAD OF TIMING CALLS
113
114             C 6/22/95 for SPEC: JWR: Initialization of TIME
115             TIME = 0
116             NCYCLE = 0
117             t1 = rtc()

```



IBM ACTC PeekPerf: Main Window

File Tools Options Action

HPM MPI SIGMA DPOMP

- Name
- [-] Probe
  - MPI\_Iprobe
  - MPI\_Probe
- [-] Recv
  - MPI\_Irecv
  - MPI\_RECV
- [-] Send
  - [-] Blocking Send
    - MPI\_Bsend
    - MPI\_Rsend
    - MPI\_Send
    - MPI\_Ssend
  - [-] Nonblocking Send
    - MPI\_Ibsend
    - MPI\_Irsend
    - MPI\_Isend
    - MPI\_issend
- SendRecv
- [-] Test
  - MPI\_Test
  - MPI\_Testall
  - MPI\_Testany

swim\_omp

Label	Count	ExcSec	IncSec
-ALL	1	0.006	1.062
-Calc1	102	0.006	0.263
-Calc2	102	0.072	0.373
-Calc3	100	0.098	0.379
-Inital	1	0.042	0.042
-Loop 100	102	0.119	0.119
-Loop 200	102	0.178	0.178
-Loop 300	100	0.209	0.209
-MPI Calc1 end	102	0.08	0.08
-MPI Calc1 start	102	0.05	0.05
-MPI Calc2 end	102	0.068	0.068
-MPI Calc2 start	102	0.055	0.055
-MPI in Calc3	100	0.072	0.072
-loop 110	102	0.009	0.009

```

calc1.f | calc2.f | calc3.f | swim_omp.f | swim_omp.f | calc1.f | calc2.f | calc3.f
101      1 0,2,MPI_COMM_WORLD,req(2),ierr)
102      CALL mpi_irecv(p(m+1,n+1),1,MPI_DOUBLE_PRECISIO
103      1 0,3,MPI_COMM_WORLD,req(3),ierr)
104      CALL mpi_irecv(vold(m+1,n+1),1,MPI_DOUBLE_PRECI
105      1 0,4,MPI_COMM_WORLD,req(4),ierr)
106      CALL mpi_irecv(uold(m+1,n+1),1,MPI_DOUBLE_PRECI
107      1 0,5,MPI_COMM_WORLD,req(5),ierr)
108      CALL mpi_irecv(pold(m+1,n+1),1,MPI_DOUBLE_PRECI
109      1 0,6,MPI_COMM_WORLD,req(6),ierr)
110      endif
111      if(taskid.eq.0)then
112      CALL mpi_isend(v(1,1),1,MPI_DOUBLE_PRECISION,
113      1 numtasks-1,1,MPI_COMM_WORLD,req(7),ierr)
114      CALL mpi_isend(u(1,1),1,MPI_DOUBLE_PRECISION,
115      1 numtasks-1,2,MPI_COMM_WORLD,req(8),ierr)
116      CALL mpi_isend(p(1,1),1,MPI_DOUBLE_PRECISION,
117      1 numtasks-1,3,MPI_COMM_WORLD,req(9),ierr)
118      CALL mpi_isend(vold(1,1),1,MPI_DOUBLE_PRECISION
119      1 numtasks-1,4,MPI_COMM_WORLD,req(10),ierr)
120      CALL mpi_isend(uold(1,1),1,MPI_DOUBLE_PRECISION
121      1 numtasks-1,5,MPI_COMM_WORLD,req(11),ierr)
122      CALL mpi_isend(pold(1,1),1,MPI_DOUBLE_PRECISION
123      1 numtasks-1,6,MPI_COMM_WORLD,req(12),ierr)
124      endif
125      if(taskid.eq.0.or.taskid.eq.numtasks-1)then
126      CALL MPI_WAITALL(12,req,istat,ierr)
127      endif
128      call f_hpmstop( 17 )
129
130      C
131      RETURN
132      END
133
134      SUBROUTINE CALC3Z
135      C
136      C          TIME SMOOTHER FOR FIRST ITERATION
137      C
138      PARAMETER (N1=513, N2=513)
139
140      include "mpif.h"
141      COMMON/decomp/js,je,taskid,numtasks,req(16),
142      1 istat(MPI_STATUS_SI
143      integer taskid,req
144      COMMON U(N1,N2), V(N1,N2), P(N1,N2),
145      * UNEW(N1,N2), VNEW(N1,N2),
146      1 PNEW(N1,N2), UOLD(N1,N2),
147      * VOLD(N1,N2), POLD(N1,N2),

```

# IBM HPC Toolkit Availability

- **IBM pSeries Servers**
  - Linux on Power
  - AIX on Power
- **All IBM Blue Gene systems**
  - Customers need to acquire additional licensing
  - Included in sitewide licenses for IBM Power systems
- **Work in Progress**
  - IBM Cell Server
  - Linux/Intel, Linux/AMD



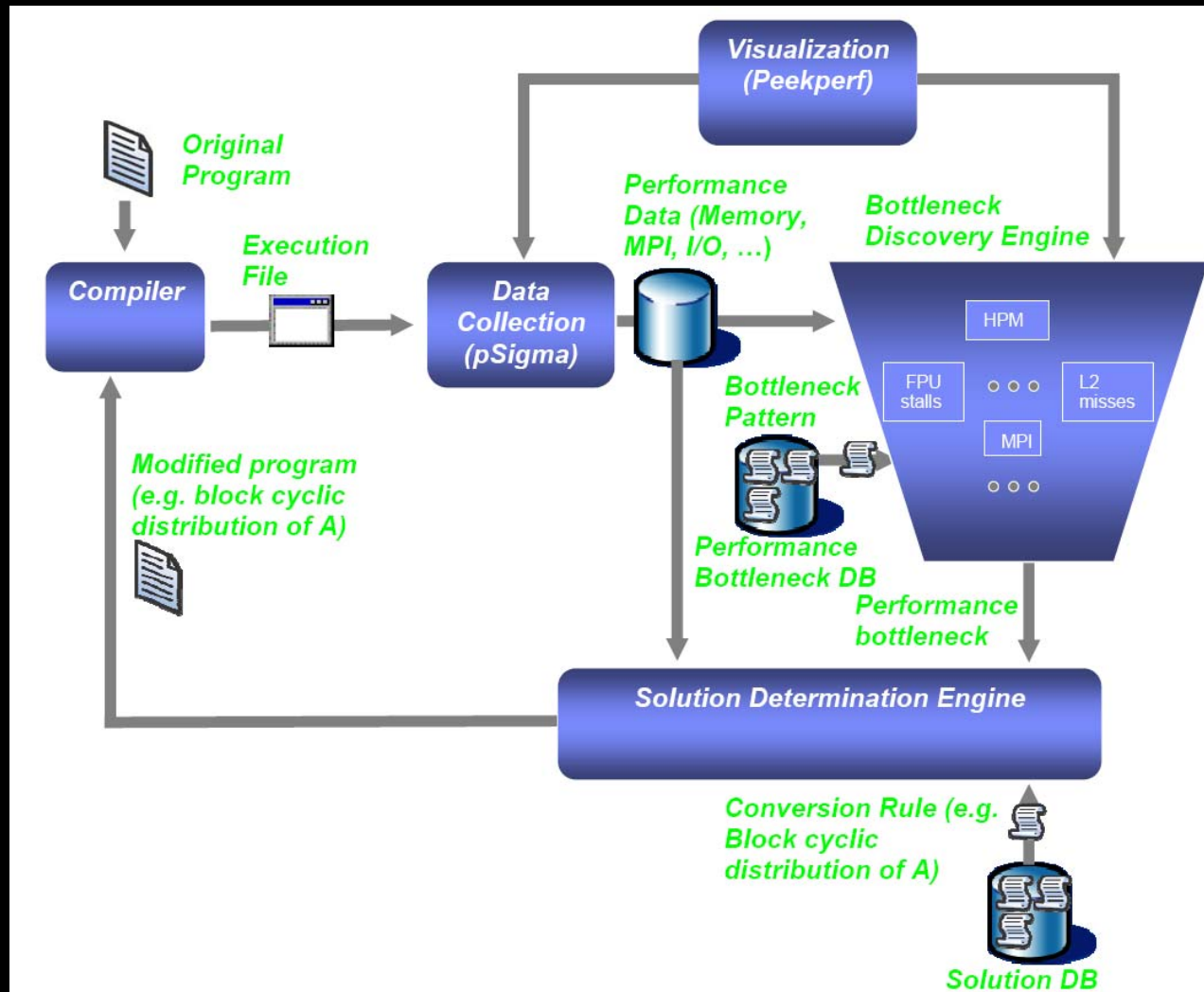
# Sanity Check on System Evolution

- **Device Scaling imposing fundamental constraints on system**
  - Power dissipation and energy consumption
  - Physical size / packaging
- **Pressure to re-think system architecture**
  - Blue Gene: low power devices, embedded (small)
  - Cell: Attached (embedded) co-processing engine
- **Systems become inherently more complex**
  - Connectivity / hierarchical topology (torus, intra-cell)
  - Memory constraints (less per processor)
  - Additional technology “boosts” (hyper-threading, SMT)
- **This poses new challenge to application programming**
  - New programming paradigm? (not on horizon - legacy codes, ISV apps, etc.)
- **Conclusion: New software tools essential to mitigate evolving system complexity**

# Automated Performance Tuning - HPCS

- **Performance Data Collection**
- **Bottleneck Discovery**
- **Solution Determination**
- **Performance Visualization**

# High Level Design for HPCS Toolkit



## Roadmap for DARPA (PERCS) Phase III

- **2007 Bottleneck Discovery Engine**
  - prototype software to demonstrate instrumentation and bottleneck determination capability
- **2008 Solution Determination Engine I**
  - prototype software to demonstrate solution determination for CPU and memory performance bottlenecks
- **2009 Solution Determination Engine II**
  - prototype software to demonstrate solution determination for MPI, and OpenMP thread bottlenecks
- **2010 Beta Release of HPCS Toolkit**
  - Full implementation of bottleneck discovery and solution determination, with user selectable source code solution implementation

## IBM ACTC long-term goals

- An **automated** performance tuning-assist framework
  - Assist users to identify performance problems
  - Provide/Suggest possible solutions
- A **common** application performance analysis environment across all HPC platforms
  - Look at all aspects of performance (communication, memory, processor, I/O, etc) from within a single interface