Infrastructure Working Group

Working Groups

- Scalable infrastructures (Jeff Hollingsworth, Al Geist)
  - Infrastructures for tool communication, data management, tracing, profiling, etc

Status

#	Challenge	Status	Future Needs
Tool communication			
	Within a single tool	MRNet, home grown	
	From the OS/machine	Papi	Fault tolerance, Network, debugging
	Reuse		
	Tool interop/data exchange		Description of metric semantics
Data management			
	measurement data and meta data from runs	PerfDB, PerfTrack,	Automated meta data collection
	Data reduction & presentation tools – graph tools		
	Other Data Sources - memory	hpm	
	Distance between user and machine – latency and data volume		

Status

#	Challenge	Status	Future Needs
Scheduler Issues			
	Tool Deployment – launch		
	Co-scheduled jobs or “spare” nodes in a job request for tools or fault tolerance		
	Topology map info, control of placement of “spare” nodes		
	Tool composability – all fighting for same hooks		
OS Issues			
	OS Scaling Issues for tools – enough sockets, ….		
	Process Control		Utrace?
	Threading Issues	Thread_db, pomp	Thread control for tools, access to thread state info
	Low overhead access to performance data	perfmon	

Status

#	Challenge	Status	Future Needs
Binary Manipulation			
	Binary Re-writing	Pin, atom, valgrind	Dyninst++ under devlopment
	Compiler Hooks	Rose, gcc instrumentation	
	Binary Analysis		
	Stack un-winder	libunwind	
Other			
	Tools survive machine faults		
	Scripting Languages		
	Tracing: Parallel filesystem, avoiding I/O	OTF	Integration of trace generator
	Profiling:		online access to data

Findings

- Tool Infrastructure reuse is uncommon, stove piped tools make this hard.
- At “peta-scale” tools must handle:
  - 100k cores soon (and up to 1M cores in the future).
  - 2 GB executable & Large number of dll’s.
  - Support multiple architectures in a single node.
- Applications & Systems will be more dynamically adaptable, and tools will need to handle this.
- Tools need communication abstractions beyond TCP/IP sockets.
- The costs of supporting tools for multiple platforms and Operating Systems is straining tool developers.
- Going to Peta-scale will increase the need for anomaly detection and (dynamic) data reduction.

Recommendations

#	Challenge	Probability	Impact
Tool communication			
	Within a single tool	Medium/High	High
	Interfaces to the OS/machine data	Medium	Medium
	Lack of Reuse	High	High
	Tool interop/data exchange	High	Medium
Data management			
	measurement data and meta data from runs	Medium	Medium/High
	Data reduction & presentation tools – graph tools	High	High*
	Availability of Additional Hardware Data – memory, etc.	Medium	High
	Distance between user and machine – latency and data volume	Medium	Medium

Recommendations

#	Challenge	Probability	Impact
Scheduler Issues			
	Tool Deployment – launch	High	High
	Co-scheduled jobs or “spare” nodes in a job request for tools or fault tolerance	High	Medium/High
	Topology map info, control of placement of “spare” nodes	Medium	Medium
	Tool composability – all fighting for same hooks	High	High
OS Issues			
	OS Scaling Issues for tools – enough sockets, ….	High	High
	Process Control	High	Medium
	Threading Issues	Medium	Medium
	Low overhead access to performance data	Medium	High

Recommendations

#	Challenge	Probability	Impact
Binary Manipulation			
	Binary Re-writing	High	High
	Compiler Hooks	High	Medium
	Binary Analysis	Medium	Medium
	Stack un-winder	Low	High
Other			
	Tools survive machine faults	High	Medium/High
	Tools for Scripting Languages	Medium/Low	Low
	Perf. Data I/O Support: Parallel Filesystem, avoiding I/O via reduction	Medium	High
	Sampling Support for Signals/Interrupts	Medium	High
	Clock Synchronization	High	Medium
	Scripting Languages for Tools	Medium	Medium/Low

Recommendations

#	Challenge	Probability	Impact
Other 2			
	Funding for Tool Infrastructure	High	High
	For useful/popular tools - Funding for full Tool life cycle – design, harden, support, maintain	High*	High