Runtime Summit - April 9, 2014
Jim Stone notes

Goals:
- Roadmap for unified runtime systems architecture for exascale
  - Challenges and solutions
  - Questions / Answers
  - Report

Top 6 Challenges
- State of the art
- Questions
- Vision

- Runtime Systems: OCR, HPX, ARTS, SEEC, GVR
- Whole machine/whole system should be represented, though most of the work is on the node. The node needs to comprehend the network.

Strawman set of challenges

Strawman set of questions
- Should we talk about interoperability?

Current runtime investments
- ARGO, HOBBES, X-ARCC projects

X-Stackwiki
- Runtimes (application-facing) table is done
- Ron's table will be done for OS/R meeting, end of April

- Research agenda will continue regardless of funding of ECI. This workshop is for the community, and for future workshops.

• Top Challenges Brainstorm
- Growing gap between communication and computation
- Scalability - underlying theme for everything; Starvation - runtime systems need to improve
- Locality & data movement
  - How static, how dynamic?
- Power management
- Overhead: impact on energy, power, time
- Resilience - due to scalability & power
  - Fault mitigation
- Resource management / load balancing
- Heterogeneity: static & dynamic, in storage and computation
- In-situ data analysis and data management. Interoperability comes into play -
  - Shared access to a memory space
- Exploitation of runtime information, introspection
  - Including in-situ analysis of performance data, self awareness
- Programmability - reuse pieces of code by runtime, i.e. composability, workflow management, i.e. workflow
- Complexity/optimization/tuning
- Portability across different systems
- Synchronization - event-driven, mutual exclusion, hierarchical
- "Computing everything" - CPU, GPU
- Name space - distributed, uniform, hierarchical, location management, data and computation
- Split this list by application-driven versus machine -drives
- Application driven needs to address tools
- Migratable stack units or threads
- Tight coupling of HW elements, part of HW support for runtime systems
- Predictive techniques for data fetching - actually more of a solution
- System management, coupled w/ OS

• General discussion
- Ron: need to define where the "runtime system" actually is, what it is
  - Executing entity
  - OS understands the system and the workload
- Thomas: A system relevance needs to be taken into account
- Wilf: need to identify services that can be dispatched as required. Think of services rather than application-facing versus machine-facing.
- Feb10 report seen by Sanjay, within node expected to be more critical, but want to push back thinking that cross-node have bigger challenges.

- What is a node? Data movement is top priority. Runtime should not be tied to the architecture of the machine.
- Policies are decided on an application basis and will vary - out of scope today- especially since they are contentious
- Focus on "within job"
- Runtime needs to support composability of application modules
- What is the new role of a compiler in the presence of a runtime system?
- Levels: intranode, enclave everything to do w/ one app) or global OS, or same as local versus global

- How to group
  - Applications / systems
  - Problem / approaches - keeping scope in mind
  - Services
    - But need to optimize around time / energy

- Grouping of Top Challenges
  1. Scalability
  2. Locality & data movement
  3. Resilience
  4. Variability, static and dynamic
  5. Heterogeneity
  6. Portability and interoperability
  7. Resource allocation
  8. Usability
  9. Composability

- Proposed runtime services - hints and requirements
  - Identify, schedule, allocate a thread (task, computation unit, work unit - code and data) to be executed by system
    - Code generation
  - Dynamic node allocation, aka compute resource allocation - interface to app
    - Including network BW
    - Give resources, release resources
  - Introspection, about power, performance, heterogeneity
    - Variability
  - Name space management; creation, data transfer, execute the units of work, translation local to global
    - Protection, isolation, security
    - OS-requested services, resource management
  - Communication & synchronization (event oriented) of data and code
  - Concurrent control
  - Location, affinity, and locality services: identify, locate , name
  - Resiliency - express error checking & recovery, to computation, data, and HW resources
  - Load balancing & scheduling , externally managed and/or automatic internally controlled
  - Migration services, data and work transformation - not separate from other services.

- Service attributes
  - How provided
  - Expected resilience
  - Expected resrouce usage
  - Persistence of memory
  - Locality attributes

- Thomas: OS may also make runtime requests, in addition to user
- Wilf: a service can request another service
- Thomas: each of the 9 teams should produce a white paper of how these challenges should be addressed.
- Want to see a common set of terminology

- How our current investments are addressing these services
  - i.e. a table of runtimes versus challenges on the wiki

- Our development is for the research. The researchers need to be able to see our, eventually one, system. But it won't be one, there will be multiple instances. But there needs to be reusability. But if we unify, we may be constrain. We need to make it easier for them to: 1) know what's going on; 2) learn from proxy apps; 3) understand taxonomy… sequence of steps. App developers don't want to have to rewrite their apps.

Wilf - Community Questions
- Research, development, production - need to push from research as soon as possible
- Need to move away from bulk synchronous - chicken/egg
- Independent foundation is important to success
- Composability is key

- Cathy
  - How does research community interact with all of the vendors, especially those not in the room?
    - § We are bringing a panel of all the vendors together
  - How do we get buy in? Have to solve a problem that they care about.
  - Apps people want portability, ability to run across all machines.
- Thomas
  - Completely prepared to sign up to Wilf's premises
  - Caveats
    - § Imperative of research
    - § Multiple codes instances in a common community
    - § Happy to have Rice to run such a forum, but there is competition with OCR and Xpress
    - § Algorithm changes won't be trivial.
- Vivek:
  - A university managing a forum is too narrow, why we need an independent foundation
- Andrew
  - Are there open minded individuals willing to contribute?
- How different from MPI Forum process?
  - Believe it is consistent
- Why a code base in a single place?
  - Need to be able to download something I can use, rather than assemble things from multiple locations.
  - Same licensing
  - Same update process
- How does it work?
  - Members decide on a project, get contributors, volunteers together, to commit to release schedule
- Who directs them, with neutrality? How to eliminate neutrality?
  - The process is open, transparent. Contributors become leaders, drive the agenda
- Rich: Quality can also be a driver, in addition to quantity. Depends on process.
- Do we need an example funded by an agency?
  - Sematech
  - Attaching ourselves to something else would be a problem, but to model based on BKMs.
- To be discussed at PI Meeting.

Key abstractions to be identified
- --- and jointly supported by the runtime system, compilers, and HW arch
- Unit of computation - task, thread
  - Attributes: locality, synchronization, resilience
- Naming: data, computation, objects that combine both (active objects)
- Global side effects; programming model abstraction?
- Execution Model
- Machine Model
  - Resources: memory, computation, storage, network, …
- Locality
- Control state: collective of info distributed across the global system that determines the next state if the machine. Distributed snapshot of the system. Logical abstraction.
- Enclave
- Scheduler: local of single execution stream, global
- Execution Stream: something that has HW associated with
- Communication data transfer
- Concurrency patterns, synchronization
- Resilience, fault model

• 4 volunteers to work offline by end of April to clean up the service definitions by 4/22, then post the matrix to be filled in by the May PI meeting - Vivek, Wilf, Rich, Kathy
- Examples
- Use key abstractions
- Team to refine by 4/22
- Then generate the matrix by end of April

Vision - Kathy
Runtime system that enables efficient execution of DOE mission applications on exascale hardware, addressing the challenges of massive concurrency, data movement minimization, increasing performance variability, energy and resource management, and tolerance of increasing failures.
- "Runtime systems" to be added
- "Dynamic adaptive" to be added, see above

Questions
- See Sonia's slides "Runtime system software architecture"

Vision components
- See Sonia's slides

- Components need to be described - Vivek, Sanjay, Thomas, Constin, Ron (composability), Vijau, Milink, Andrew

Vision interface
- Each person will be looking at each component

Vision: measure of success
- Microbenchmarks are a good place to start, but can be misleading. Mini-apps are better.
- Number of users

Roadmap for Runtime Research
- New runtime research
- Expected summit outcome
- Decision: dynamic or not? If dynamic, then when?
- Major envisioned milestones and timeline

Thomas: efficiency improvement and scalability need to be understood to get to exascale.

Runtime System workshop may be in August. Need to determine good time to have, let Sonia know by 4/16.