PERIXML Goals

- Initial Starting Point:
  Storage of metadata from performance experiments
  - Which binary/libraries, when, on which machine/nodes?
  - Import into performance databases
  - Link to performance data

- Need more: store actual performance data

- PERIXML as tool exchange format
  - Easy data exchange and tool interoperability
  - Support for profile data (not traces)
  - Raw and processed data
Basic Format: Five Profile Dimensions

- Each data point has a connection to any dimension
  - Code: code locations (binary, source)
  - Time: timeline
  - Space (?): nodes, tasks, ranks, communicators
  - Metrics: which data to collect
    - Metric algebra for derivations
  - Dynamic state (call stack, context, …)

- Each dimension can have multiple hierarchical subdivisions
  - What does this mean for metrics?
  - How to define groups?

- Aggregations
**Open/Future Questions**

- How to represent aggregation?
- How to deal with MPMD applications?
  - Impact on metadata (represent multiple binaries)
- Call stack representation
- Don’t worry about for now:
  - Connections between hierarchical subdivisions
    - Dynamic/Static Code Locations
    - Nodes/Tasks/Threads – Ranks/Communicators
  - Constraints/Relationships between dimensions
- Store structural information (e.g., CFGs)
  - Connection to OpenAnalysis (?)
  - Store binary analysis results (e.g., from Dyninst)
XML Format

- Seven sections with separate tags
  - MetaData
  - CodeDimension
  - TimeDimension
  - SpaceDimension
  - StateDimension
  - Metrics
  - Data
XML Tags: Metadata

- Reuse existing XML Spec
- Remove duplicates (compared to next sections)
- Support MPMD
XML Tags: Code Dimension

CodeDimension
  Application <name>
    Binary/Library <name>
      Function <name> [<arguments>]
        Loop (?)
          PC <addr>
  SourceFile <name>
    Function <name> [<arguments>]
      Loop (?)
        Basic block <number>
          Source line <line>

-OR-
  Region <type> <name> [<argument names>]
    Region <type> <name>

  types={file, functions, loop, basic block, line, …}
XML Tags: Time Dimension

TimeDimension
  Interval <start> <end>
  Interval
  Interval
  ...

Any interval based on code state/regions, e.g., iterations, Should be part of the state dimension
XML Tags: Space (?) Dimension

SpaceDimension
  Nodes
    Tasks
      Threads
  Ranks
  Communicator
    Rank
    Rank
    ...

Group Definitions?
XML Tags: Metrics Dimension

MetricsDimension
  Metric <name>
    Description <name>
    Operation */|/+|- |
      Metric 1
      Metric 2
    Type/Unit (allow only one type)
      Time | State | Count

Do we really need hierarchies for this dimensions?

Derived metrics using algebra
  describe using references (as illustrated above)?
  describe using a hierarchy (easier, but could lead to replication)?
How do we describe arbitrary numbers of operands?
Include min/max?

Include description on how the data was collected
XML Tags: State Dimension

Answers the question: How did I get here?

StateDimension
  State <type> <name> [<argument>]
    State <type> <name> [<argument>]
      State <type> <name> [<argument>]

  type = {phase, iteration, function, callstack, callpath depth}

StateCombination <name>
  Operator = AND, OR, NOT
    (anything else?)
  State1
  State2 …
State Example

- Function main
  - Phase computation
    - Function Bar (x = 5)
      - Iteration 3 (main loop)
        - 2nd iteration of convergent solver
          - Loop
            - block
              - Line
            - Iteration 4
        - ...

- Region (name, type, value)
  - region

Can define new states as Set of other states
XML Tags: Data Point

Data
  Code <code location> <aggregation>
  Time <time location> <aggregation>
  Space <space location> <aggregation>
  Metric <metric name>
  State <state name>
  Value <val>

Allow only one statement for each dimension

- If code, time, or space dimension is not present, then use <all> <sum> as the default
- Metric must be specified
- If state dimension is not present then measurement refers to any state