Infra SIG Update:

What is new?

- Training support (Preview)
  - TrainingInfoProto (state-variables, initialization-step, training-step)

- Function update
  - Functions with context-dependent function-body
  - Functions dependent on specific operator sets

- ONNX-MLIR (in progress)
  - ONNX dialect in MLIR

Training Support

- https://github.com/onnx/onnx/blob/master/docs/IR.md#training-related-information
- Weights to be trained (a subset of initializers)
- Initialization (of weights) described using:
  - a Graph
  - a binding (map from weights to outputs of graph)
- Training step (updates of weights) described using:
  - a Graph
  - a binding (map from weights to outputs of graph)
- Gradient operator
- GraphCall operator

Infra SIG Update:

What next?

- Recurring Tradeoff:
  - Expressiveness (new ops for new models) vs.
  - Efficiency (e.g., exploit hardware features) vs.
  - Development Cost
- Better use of the function mechanism that is intended to target this tradeoff
  - Identify a better core set of primitive ops (leverage learning from multiple implementation frameworks, including MLIR)
  - Reflect the design in dialect design in ONNX-MLIR
  - Do we need more than 2 levels in ONNX?

Function Extension: Details

- Operator registration APIs extended to allow:
- Function body that depends on statically available context (attribute values, etc.)
  - OpSchema& SetContextDependentFunctionBodyBuilder (ContextDependentFunctionBodyBuilder);
  - Examples: SoftmaxCrossEntropyLoss, NegativeLogLikelihoodLoss
- Functions that rely on multiple external operator sets.
  - OpSchema& FunctionBody(const std::vector<NodeProto>& func_nodes, const std::vector<OperatorSetIdProto>& opsets);

Infra SIG Update:

Call for actions/contributions

- Backend scoreboard: please register your backends here:
  - https://github.com/onnx/backend-scoreboard
- Tools for checking compliance (IR and opsets)
  - Stricter onnx checker
  - Better test coverage with node/model level test cases.
  - Better testing for functions
- Improve Build/Setup – first user experience improvement
- Improve release process
- Improve CI
- IR levels design and implementation and ONNX-MLIR
  - identify core ops and op-categories that help simplify a backend implementing ONNX