2022 Workshop
May 24-26th

Portability and Composability

Venkat Pullela - Keysight
Surendra Anubolu - Broadcom
Shyam Kaluve - OpenNets
Madhu Dhavala - OpenNets
Sahil Gupta - RIT

Agenda

Modularity / Portability / Composability

Applets and Proposal

Merge Algorithms:

- Insertion Merge
- Feature Merge
- Field Merge

Modularity

No explicit specification of modularity constructs in P4 or NPL

Implicit use of Control Blocks in P4, functions in NPL

#include files are used to organize code

Control blocks are first choice as they specify Lookups, Logic, Control Flow

  P4 Tables are second choice

Mechanism to leverage modularity is missing, a.k.a composability

Not completely adequate for specifying a full fledged features and interactions

Portability

ASIC pipelines have too many architecture specific implementation details

- Packet forwarding, modification, replication

Portable Switch Architecture (PSA), Portable NIC Architecture (PNA) help

- Easier to adopt for Fetch-Decode-Execute architectures than ASICs

Programming models of ASICs differ

- Ability to modify packet in ingress, egress pipeline vs. at the end
- Re-create the packet vs. modify packet
- Error checking, checksum, location of state

Composability

Degree to which components can be combined with ease to build a modular system

Makes it easy, faster to build, deploy and operate large, complex systems

uP4 - Higher level abstraction on top of P4 that is translated into target specific P4 code

Lyra one-big-pipe - Multi system composability that gets compiled to native language

DAPIPE, P4 Weaver - Insertion based composability

Applets proposal

Previous work does not address feature interoperability and interactions

Leverages Conventions, Standard Models, Design patterns, Frameworks

Base pipeline - Designed as a solution to portability

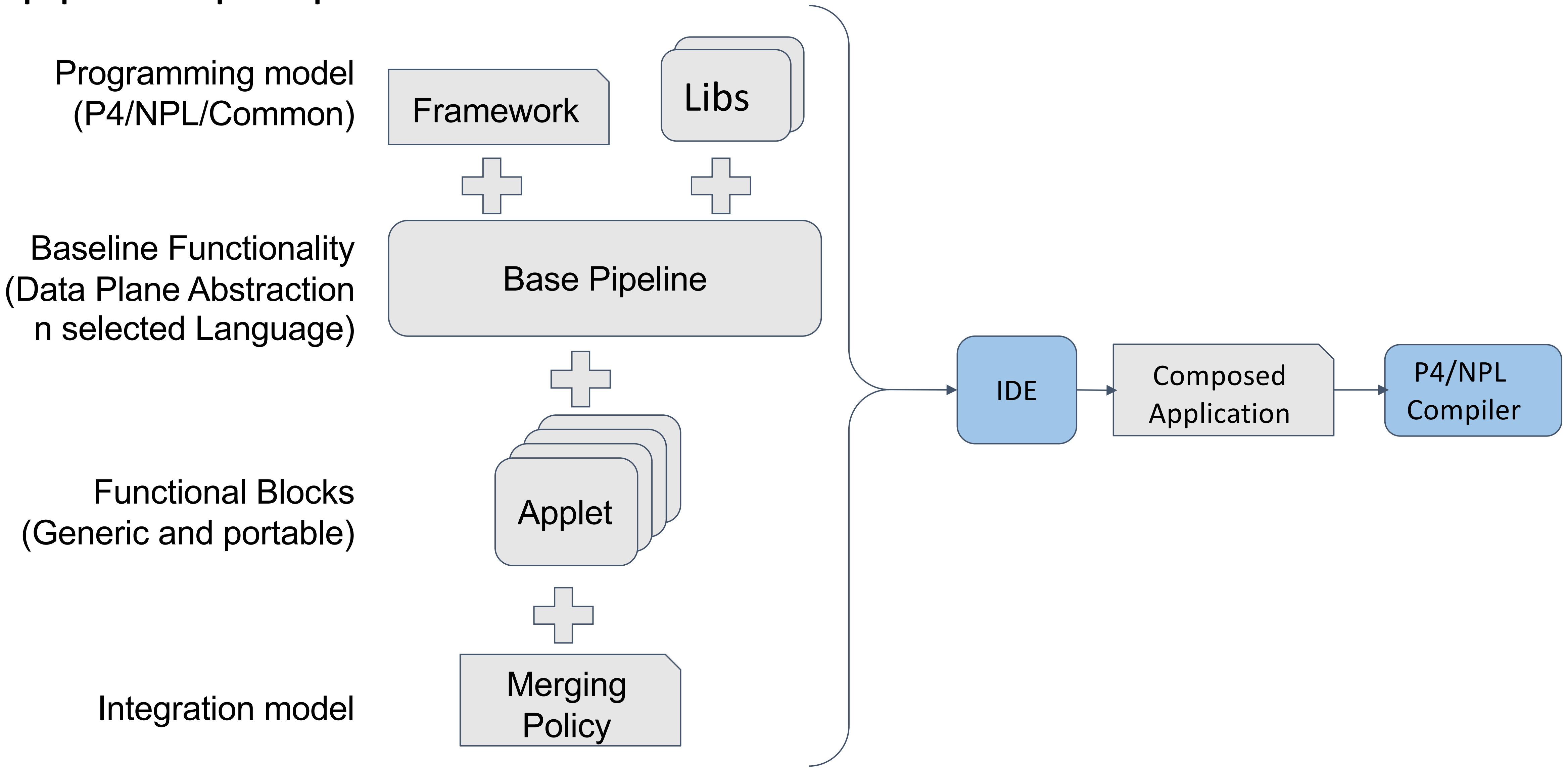
- Abstract system level services, with well defined interfaces

Modules - A separate Applet for each feature

Merge Policy - Specifies how base pipeline and Applets are combined

- Common Intermediate Representation for multilingual support
- Different models at different stages of integration

Applets proposal



Merge Algorithms

Insertion Merge

- Program/Framework explicitly defines how the applets interact and produce the desired result
- Custom merge logic is hard coded in the program in different places (except in Applets)

Feature Merge

- Describes the features that will be merged to produce a common result
- Describes the final result for each combination of the results from the Applets
- Ordered Policy rules (ACL like) describe how features are merged
- There could be multiple Feature merge sets

Field Merge

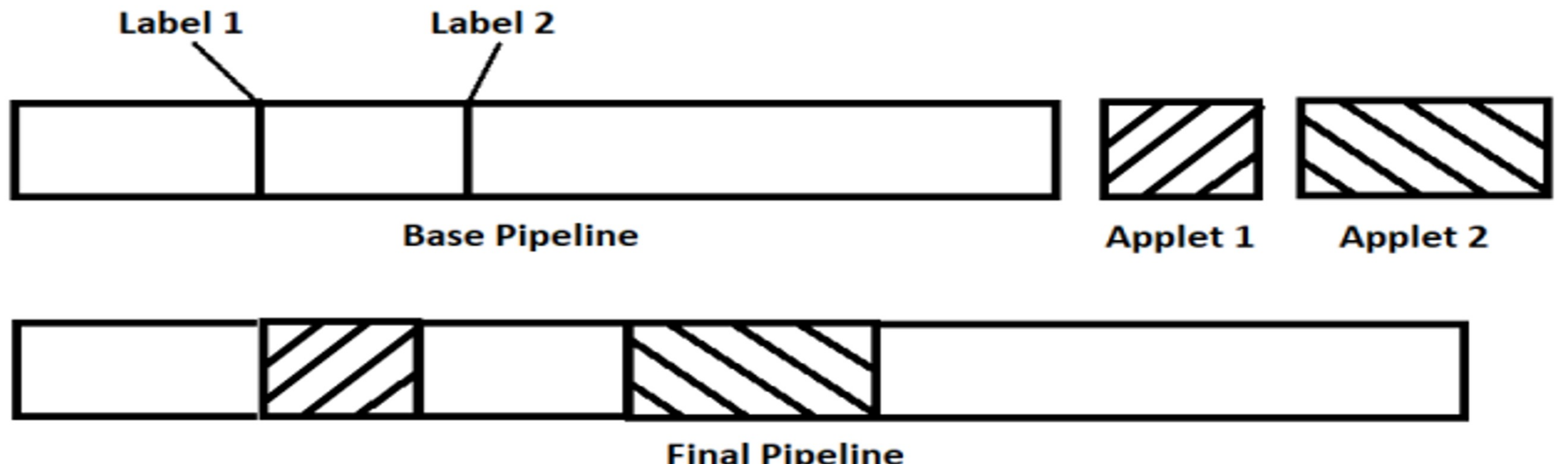
- Each Applet generates multiple result field objects
- Same result field object may be generated by multiple Applets
- Overlapping Result fields are combined based on priority or policy

Insertion Merge

Simplest and easiest way to implement modularity and composability

- Merge using, C preprocessor and #includes
- Aspect Oriented Programming (AOP)

  locate → label → insert
- Merge policy specifies what Applets are inserted, and where in the pipeline



Feature Merge

Uses Object Oriented Programming concepts

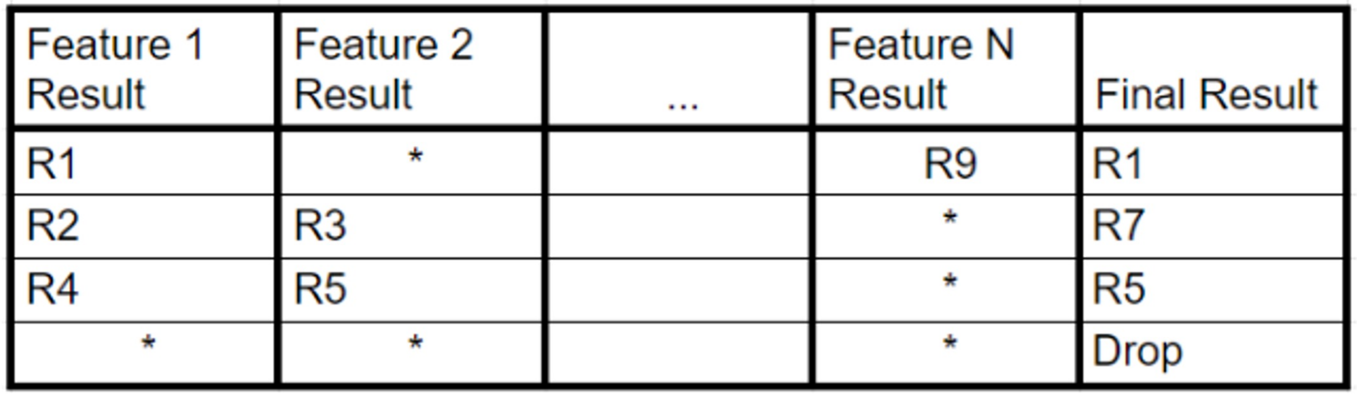
Feature Object → Input/Output fields

→ Result fields

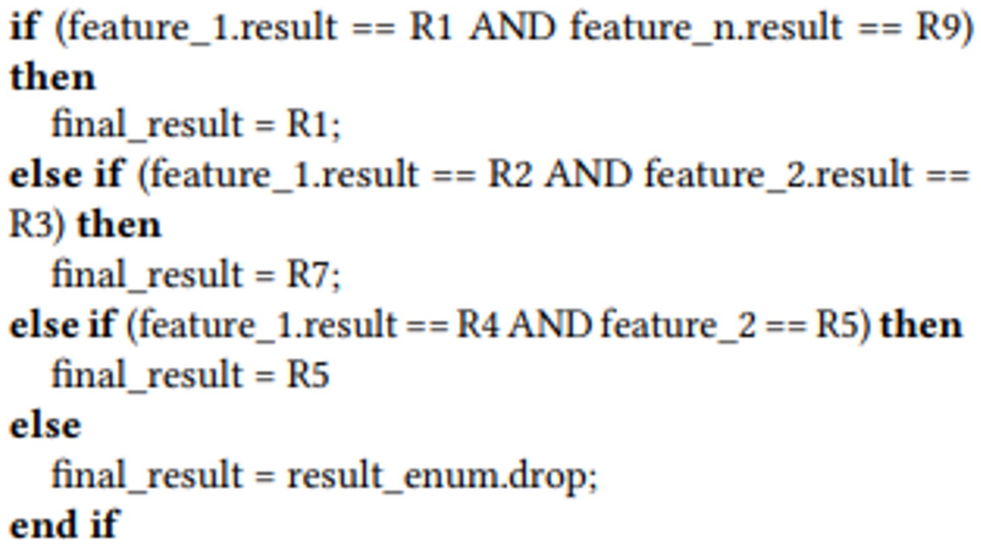
Features may specify dependencies to enforce ordering

Input, output specify order implicitly

Merge policy specifies feature interactions



Feature 1 Result	Feature 2 Result	...	Feature N Result	Final Result
R1	*		R9	R1
R2	R3		*	R7
R4	R5		*	R5
*	*		*	Drop



```
if (feature_1.result == R1 AND feature_n.result == R9)
then
   final_result = R1;
else if (feature_1.result == R2 AND feature_2.result ==
R3) then
   final_result = R7;
else if (feature_1.result == R4 AND feature_2 == R5) then
   final_result = R5
else
   final_result = result_enum.drop;
end if
```

Field Merge

- Field Merge is the fine grain merge of Applets into a single program at individual field level.
- When multiple features are combined, not all fields have a global semantic scope nor relevance across features.
- Broadcom has implemented a priority scheme in their hardware architectures, called strength i.e strength constructs in NPL
- Strength is one of the attributes of the Result object of a dis-aggregated Match Action table, along with name, size, type, scope etc.
- Strength values can be assigned at table level or individual entry level, giving a fine grain control.
- Example: System ACL rules > VRF-specific routes > PBR > Global Routes

Thank You

<additional resources>