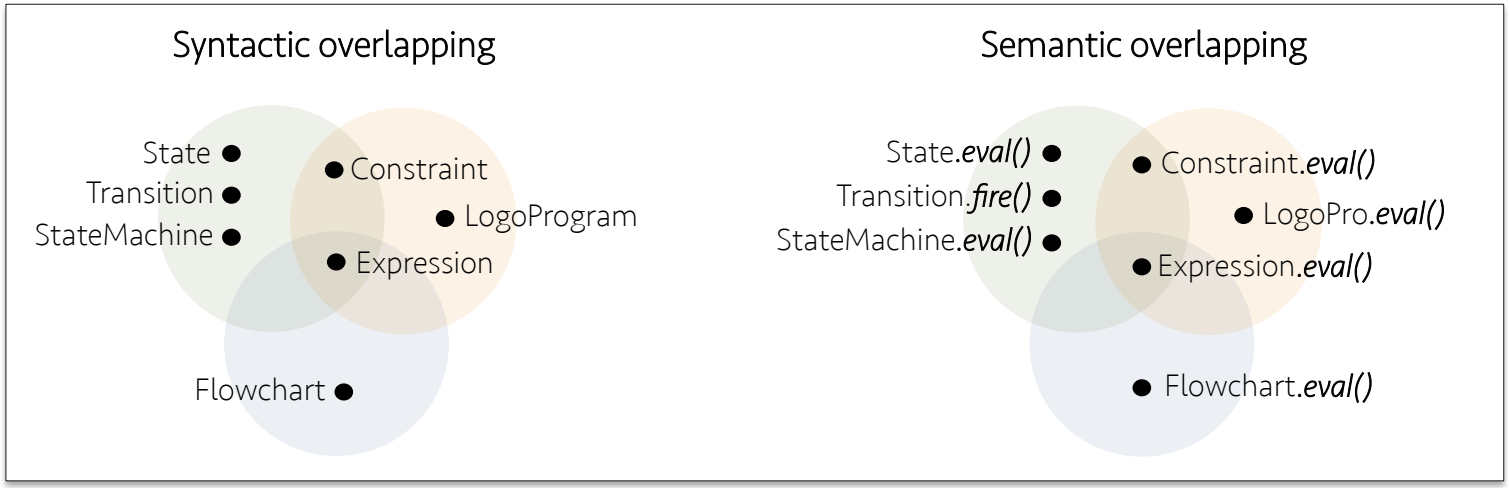
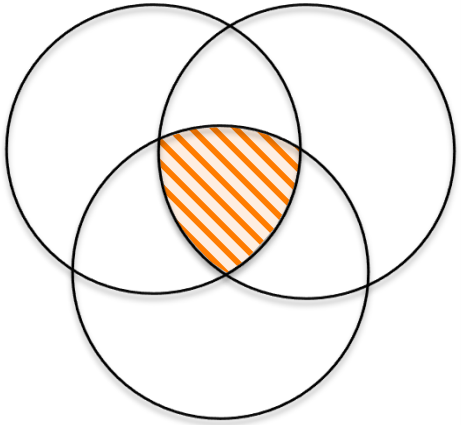
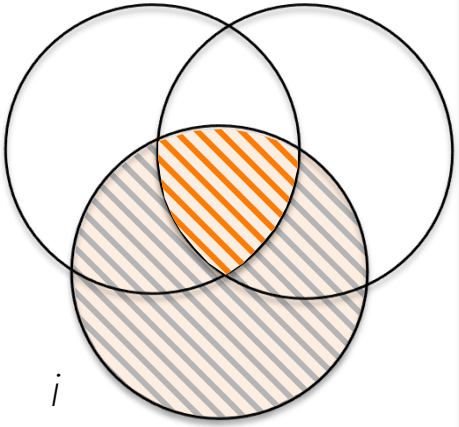
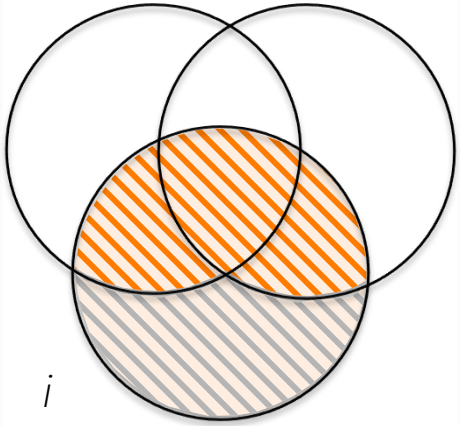
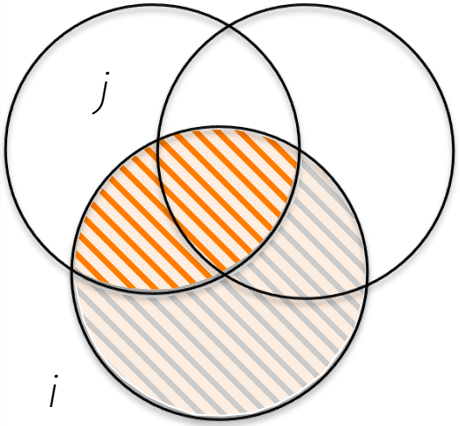
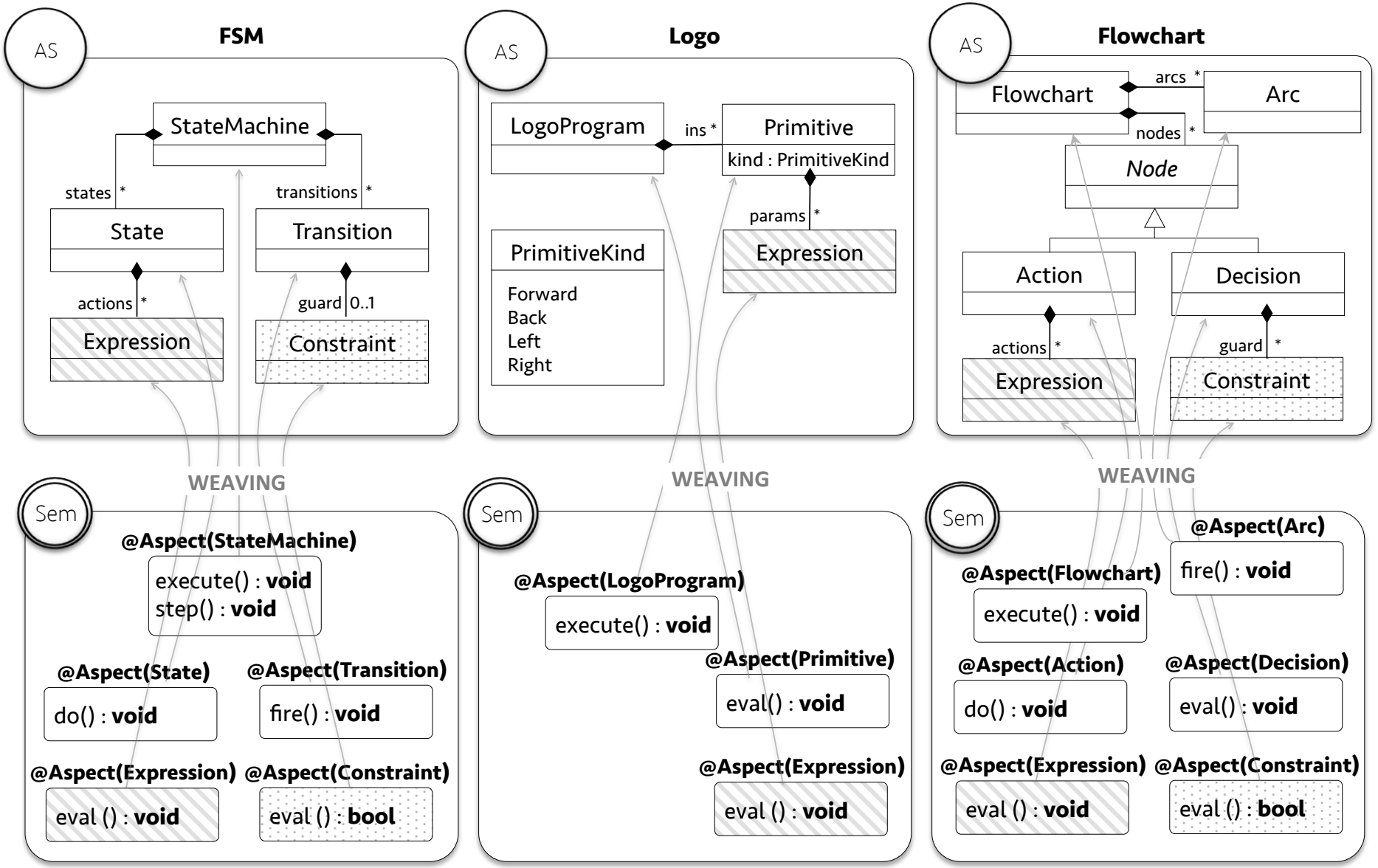
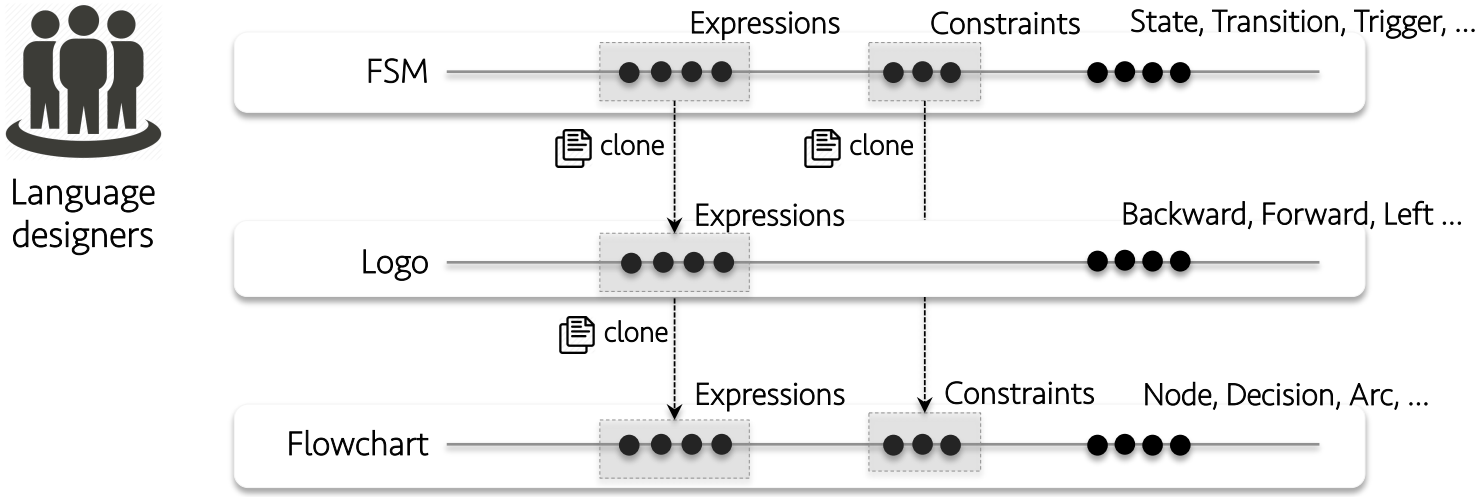
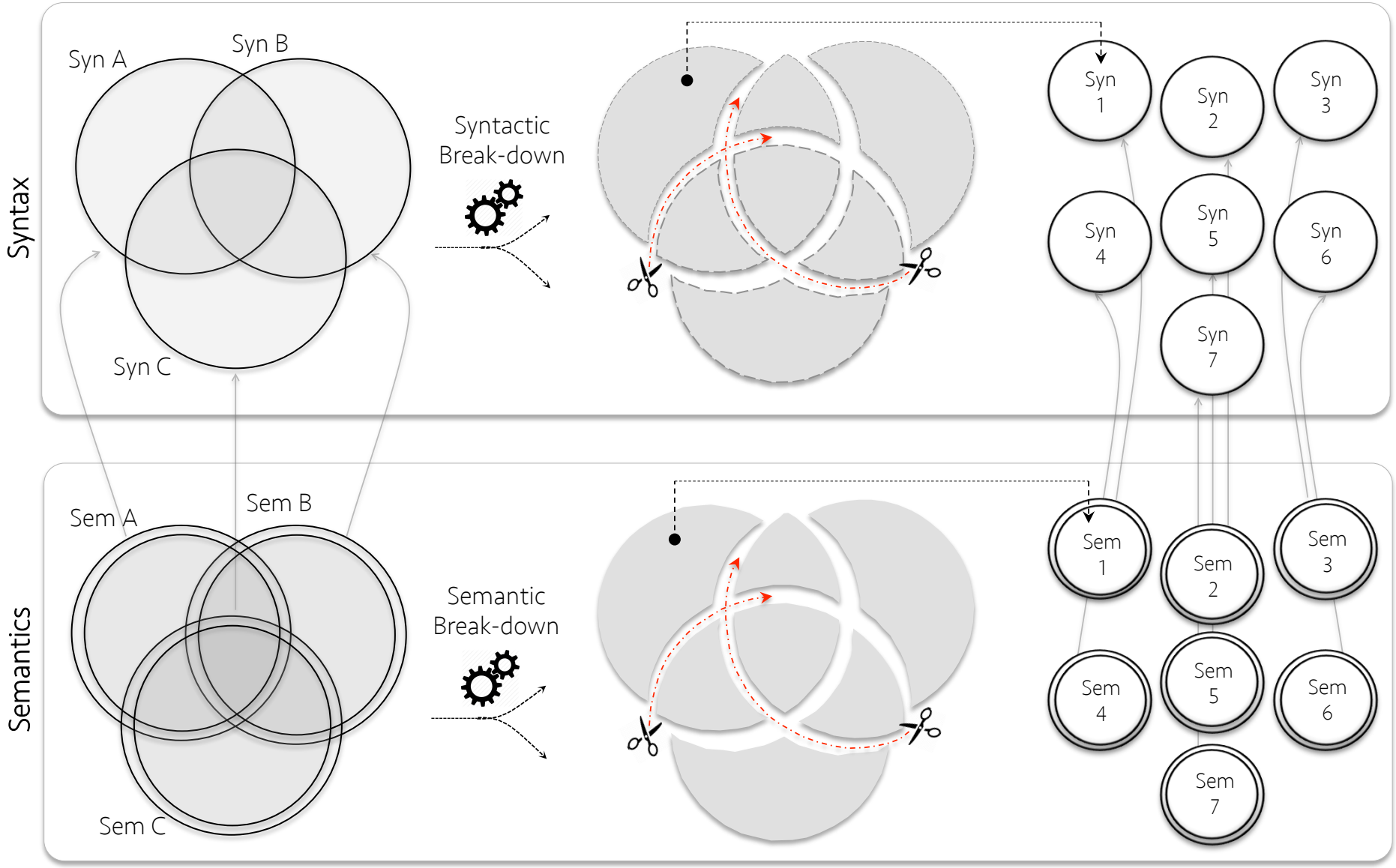

Syntactic overlapping
State
Transition
StateMachine
Constraint
LogoProgram
Expression
Flowchart
Semantic overlapping
State.eval()
Transition.fire()
StateMachine.eval()
Constraint.eval()
LogoPro.eval()
Expression.eval()
Flowchart.eval()

Size of Commonality (SoC)	Product Related Reusability (PRR_i)	Individualization Ratio (IR_i)	Pair-wise Relationship Ratio ($PWRR_{(i,j)}$)
			


AS
FSM
StateMachine
states *
transitions *
State
Transition
actions *
guard 0..1
Expression
Constraint
AS
Logo
LogoProgram
ins *
Primitive
kind : PrimitiveKind
params *
Expression
PrimitiveKind
Forward
Back
Left
Right
AS
Flowchart
Flowchart
arcs *
Arc
nodes *
Node
Action
Decision
actions *
guard *
Expression
Constraint
WEAVING
WEAVING
WEAVING
Sem
@Aspect(StateMachine)
execute() : void
step() : void
@Aspect(State)
do() : void
@Aspect(Transition)
fire() : void
@Aspect(Expression)
eval () : void
@Aspect(Constraint)
eval () : bool
Sem
@Aspect(LogoProgram)
execute() : void
@Aspect(Primitive)
eval() : void
@Aspect(Expression)
eval () : void
Sem
@Aspect(Arc)
fire() : void
@Aspect(Flowchart)
execute() : void
@Aspect(Action)
do() : void
@Aspect(Decision)
eval() : void
@Aspect(Expression)
eval () : void
@Aspect(Constraint)
eval () : bool

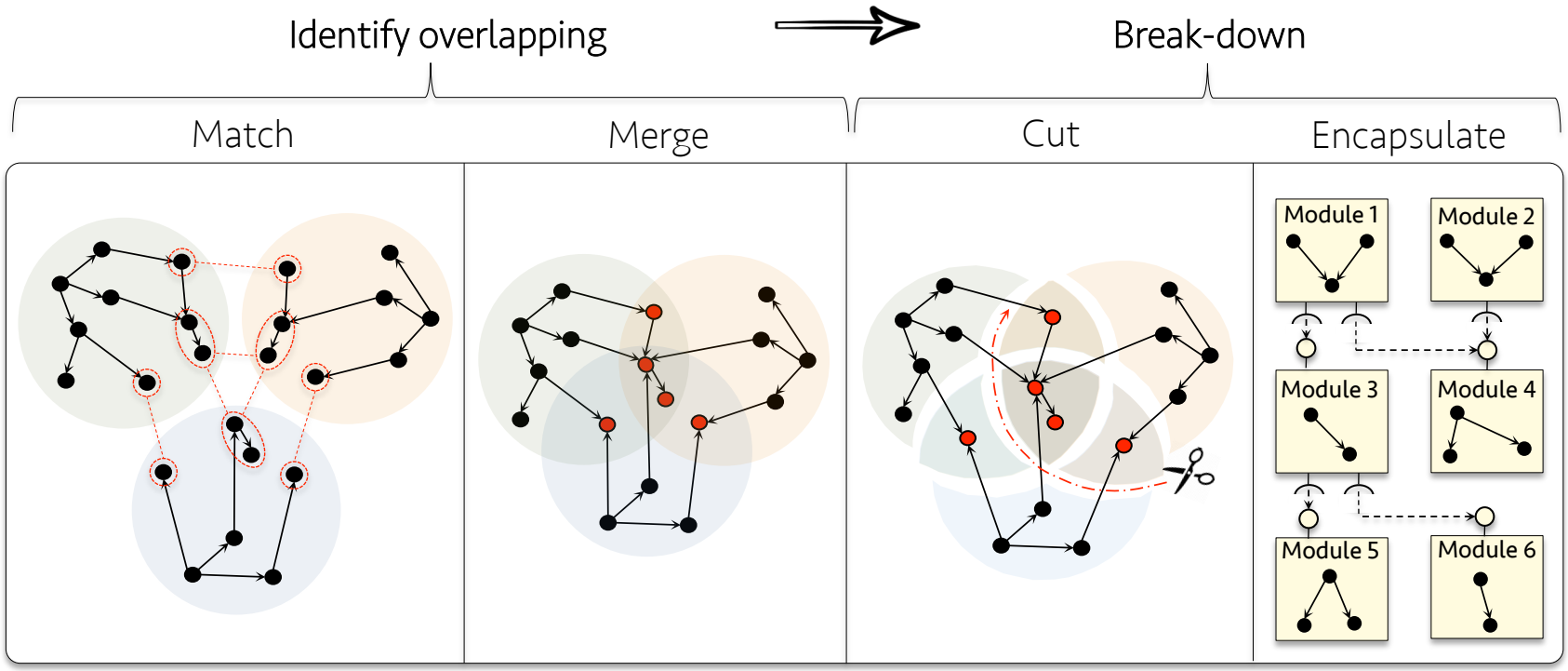

Language designers
Expressions
Constraints
State, Transition, Trigger, ...
FSM
clone
clone
Expressions
Backward, Forward, Left ...
Logo
clone
Expressions
Constraints
Node, Decision, Arc, ...
Flowchart



First: Identifying overlapping

Second: Breaking-down the input set

Third: Encapsulating language modules

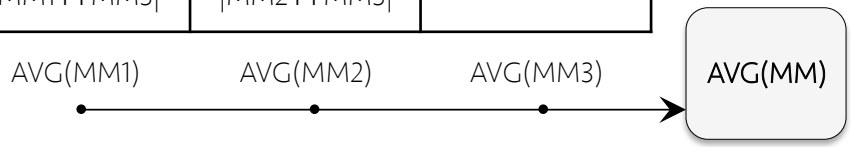


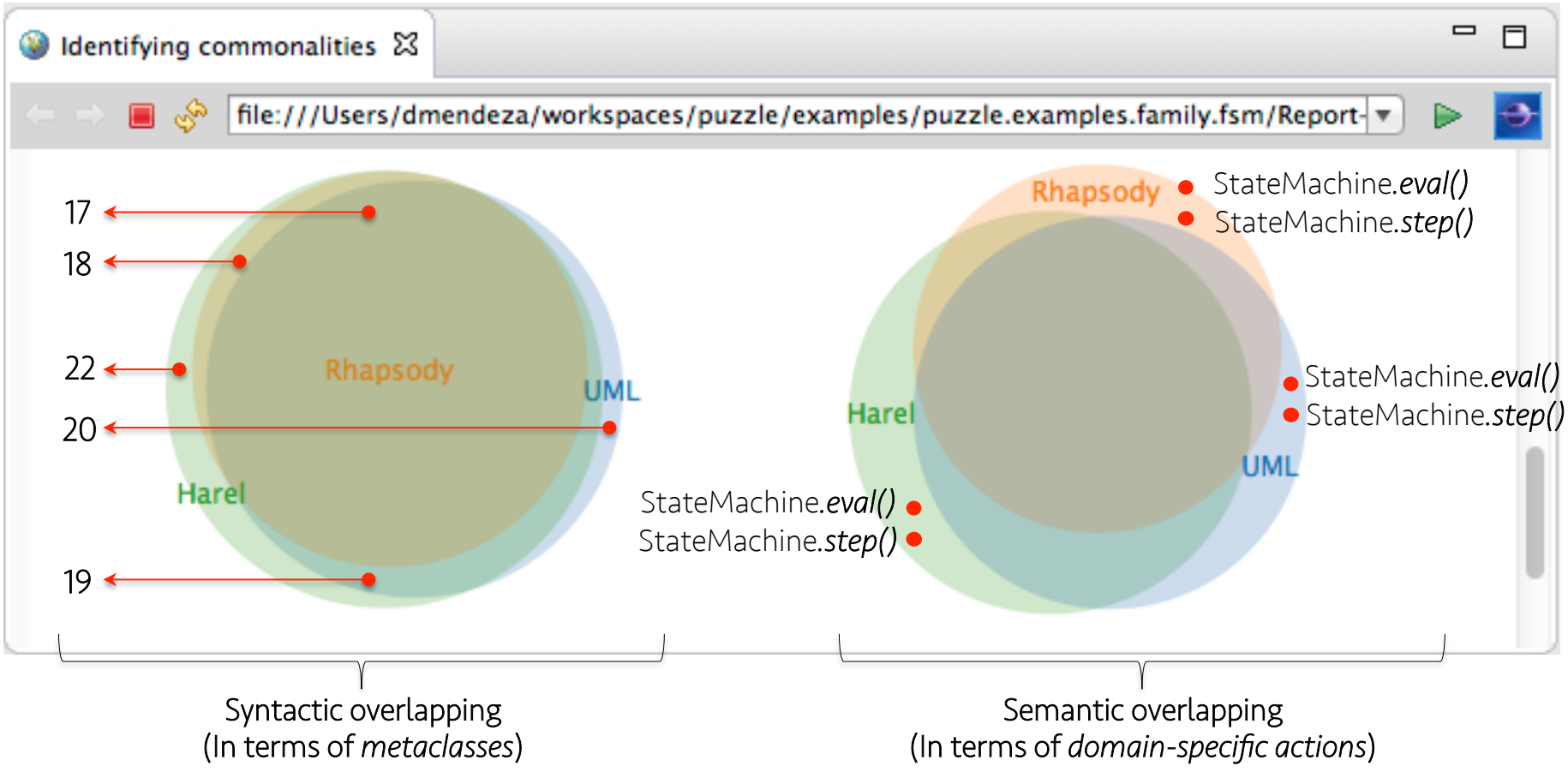
Second: Merge

Second: Breaking-down the input set

Third: Encapsulating language modules

	MM1	MM2	MM3
MM1	-	$\lvert MM2 \cap MM1 \rvert$	$\lvert MM3 \cap MM1 \rvert$
MM2	$\lvert MM1 \cap MM2 \rvert$	-	$\lvert MM3 \cap MM2 \rvert$
MM3	$\lvert MM1 \cap MM3 \rvert$	$\lvert MM2 \cap MM3 \rvert$	-




Identifying commonalities
file:///Users/dmendeza/workspaces/puzzle/examples/puzzle.examples.family.fsm/Report-
17
18
22
20
19
Rhapsody
UML
Harel
StateMachine.eval()
StateMachine.step()
Rhapsody
StateMachine.eval()
StateMachine.step()
Harel
UML
StateMachine.eval()
StateMachine.step()
Syntactic overlapping
(In terms of metaclasses)
Semantic overlapping
(In terms of domain-specific actions)