Classifying and Comparing Attribute-Based and Relationship-Based Access control

by

Tahmina Ahmed, Ravi Sandhu and Jaehong Park

ACM CODASPY

March 22-24, 2017

Outline

- Introduction
- Background & Motivation
- Attributes: Definitions and Assumptions
- ReBAC Classification
- ABAC Classification
- Multilevel Relationship Expression With Attributes
- Comparison: ABAC Vs. ReBAC
- Conclusion

ReBAC

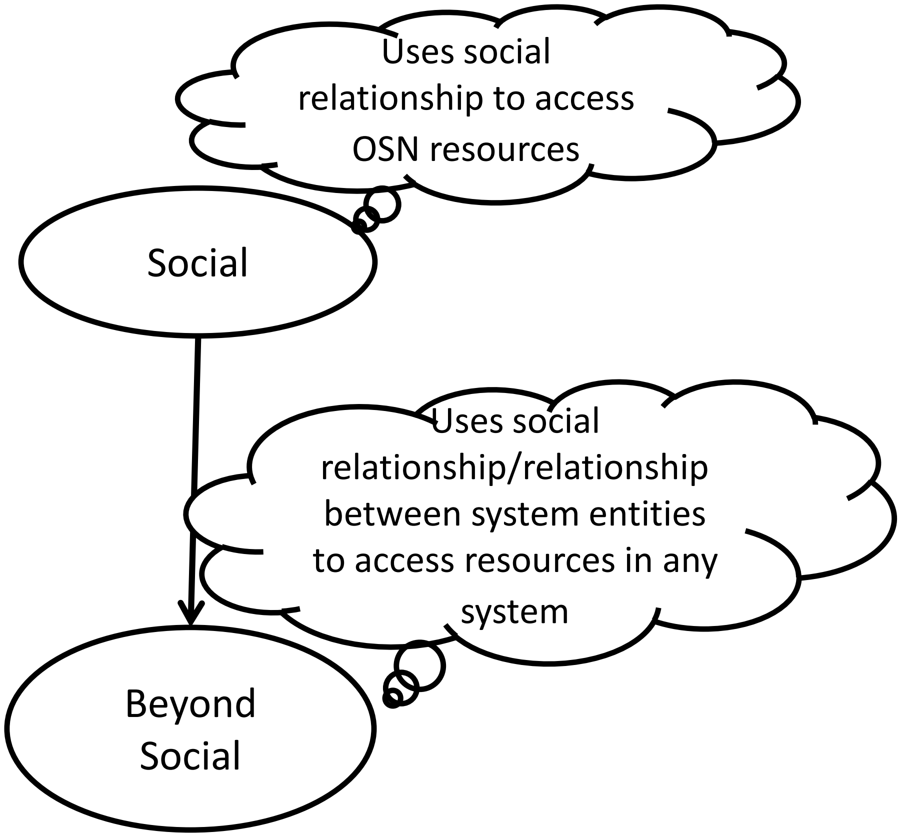

Uses social relationship to access OSN resources
Social
Uses social relationship/relationship between system entities to access resources in any system
Beyond Social

Figure 1: Using Relationship in Authorization policy expression is used for social and beyond social environment

ABAC

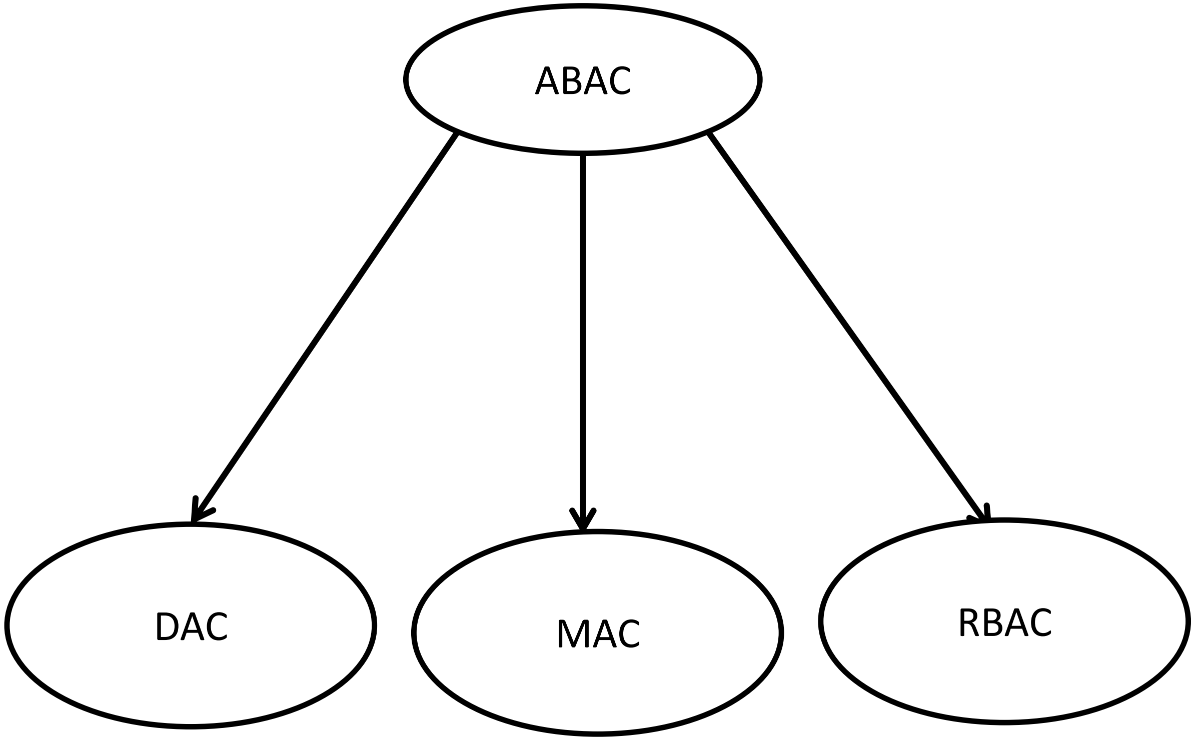
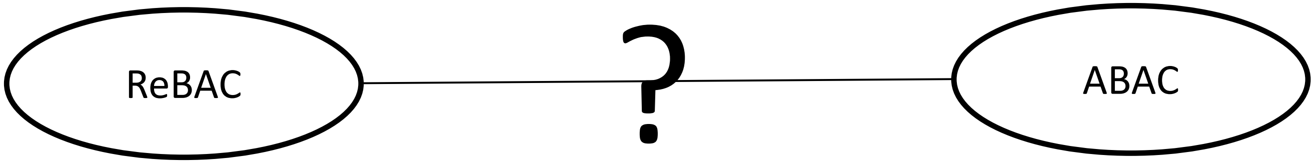


Figure 2: ABAC can configure DAC, MAC and RBAC [Zin et al. 2012]

ReBAC Vs. ABAC



- Are they Comparable ?
- Can Attributes Express Relationships?
- Can ReBAC Configure ABAC? Vice versa?
- Do they have equal expressive power?

If not

- Which one is more expressive?

Attribute Types

1. Attribute Value Structure
   - ❑ Atomic-valued or Single-valued Attribute (e.g. gender)
   - ❑ Set-valued or Multi-valued Attribute (e.g. phoneNumber)
   - ❑ Structured Attribute (e.g person-Info (name, age, phoneNumber))
2. Attribute Value Scope
   - ❑ Entity Attribute (e.g. friend)
   - ❑ Non-entity Attribute (e.g. age)
3. Boundedness of attribute range
   - ❑ Finite Domain Attribute (e.g. gender)
   - ❑ Infinite Domain Attribute (e.g. time)
4. Attribute association
   - ❑ Contextual or Environmental Attribute (e.g. currentTime)
   - ❑ Meta Attribute (e.g. role(user) = manager , task(manager) = supervise)
5. Attribute mutability
   - ❑ Mutable Attribute
   - ❑ Immutable Attribute

Attribute Function Composition

$$f : X \rightarrow Y$$

$$g : Y \rightarrow Z$$

$$x \in X \, , g\left(f\left(x\right)\right) \in Z$$

Assumptions

- All non entity attribute are finite domain
- Entity attribute functions are partial functions defined on existing entities only
- Inner attribute function in an attribute function composition should always be entity attributes
- Structured attribute is a multivalued tuple of atomic or set-valued attributes. So it is more expressive than atomic or set-valued attribute.

ReBAC Classification

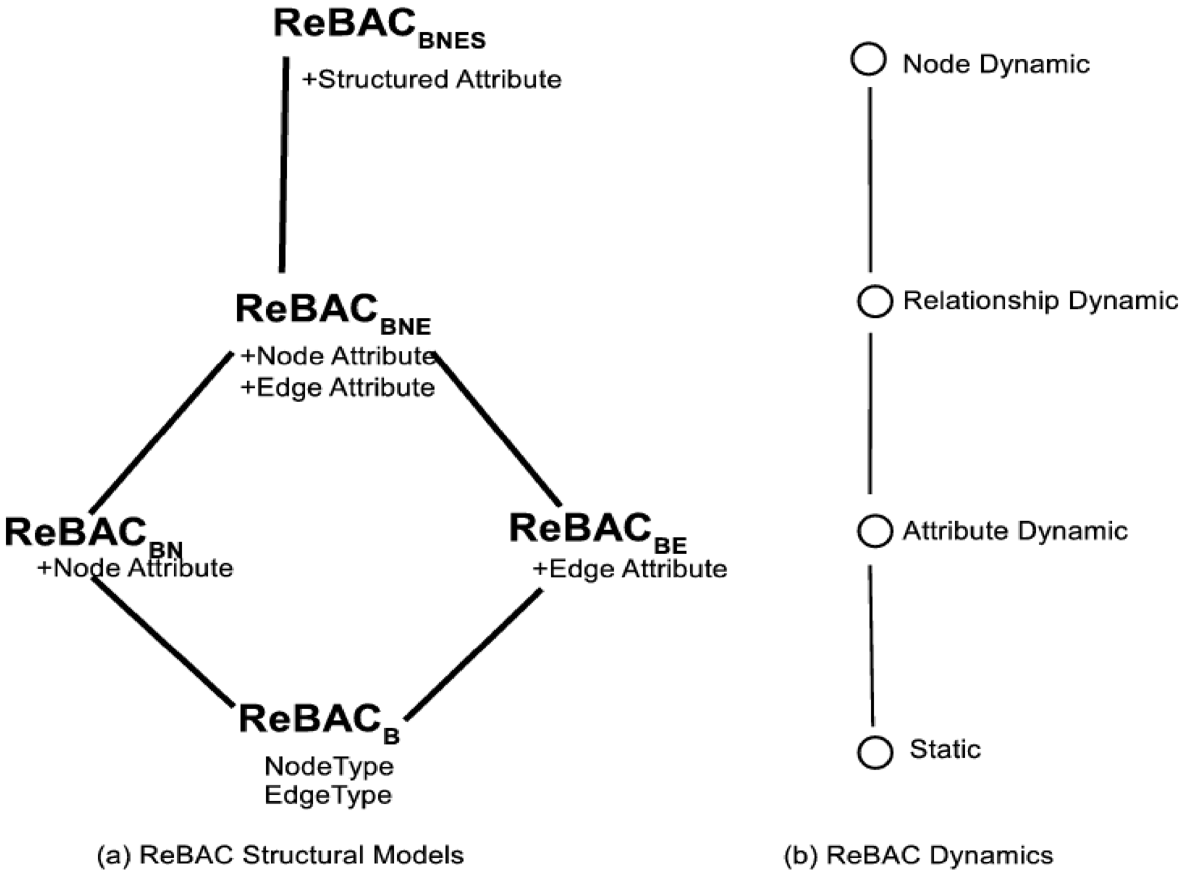


Figure 3.: ReBAC Framework

Example

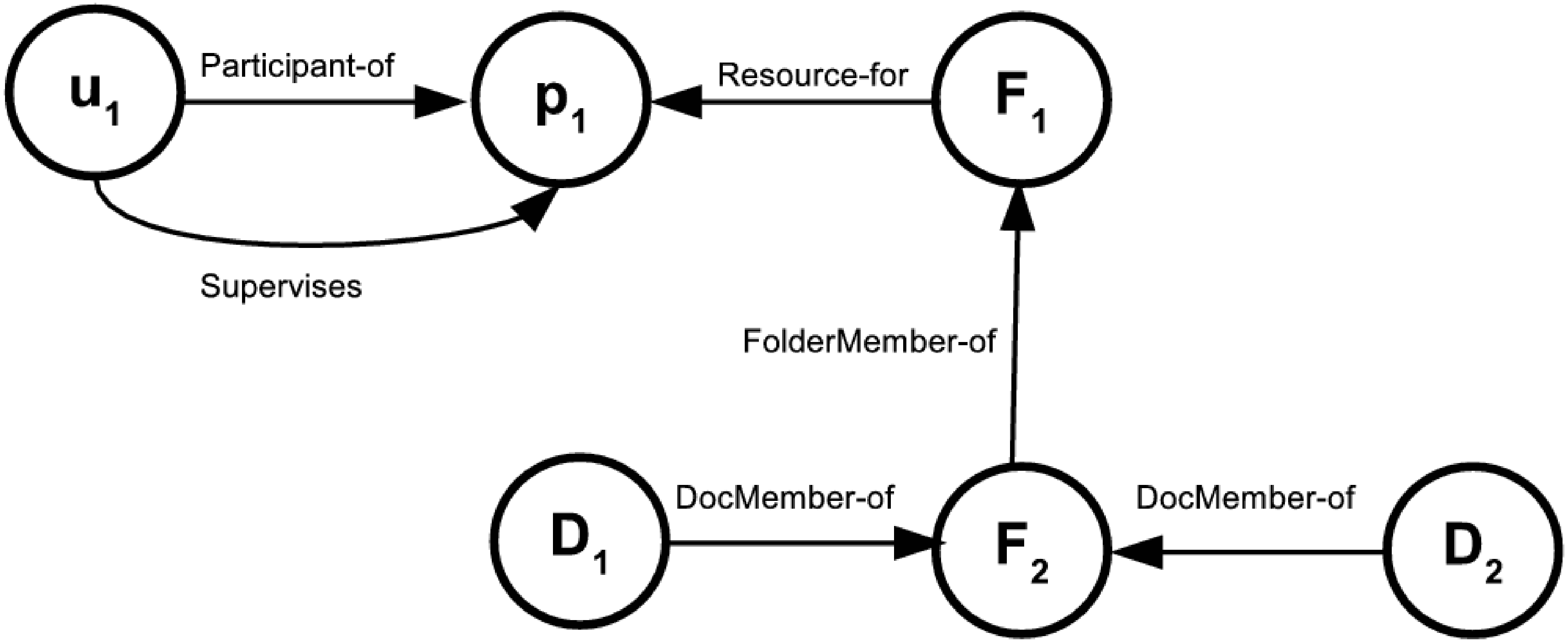


Figure 4.: A Simple Relationship Graph Expressible in ReBAC$_B$ [Crampton et al. 2014]

Example (Continued…)

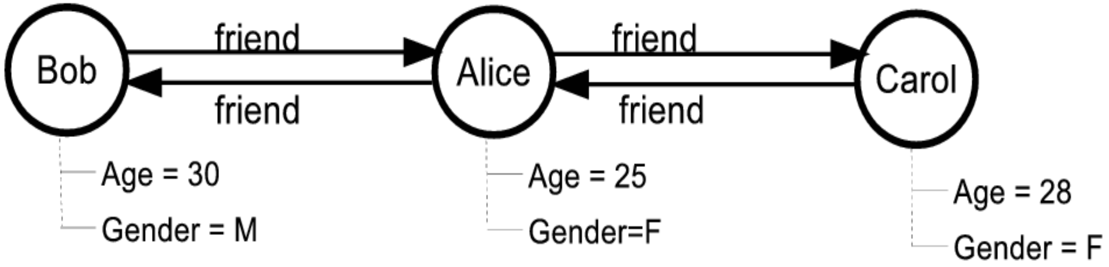


Figure 5: An Example of Node Attributes in Relationship Graph Expressible in ReBAC$_{BN}$

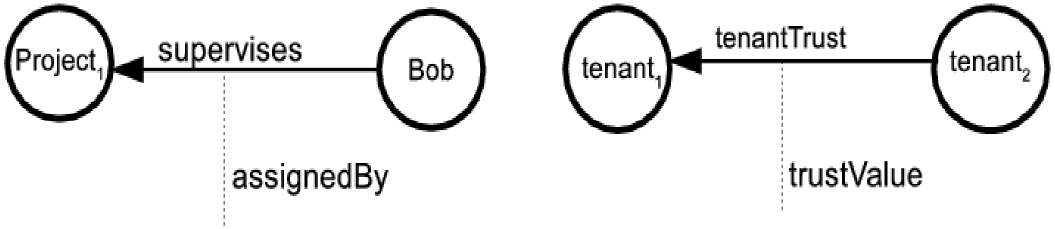
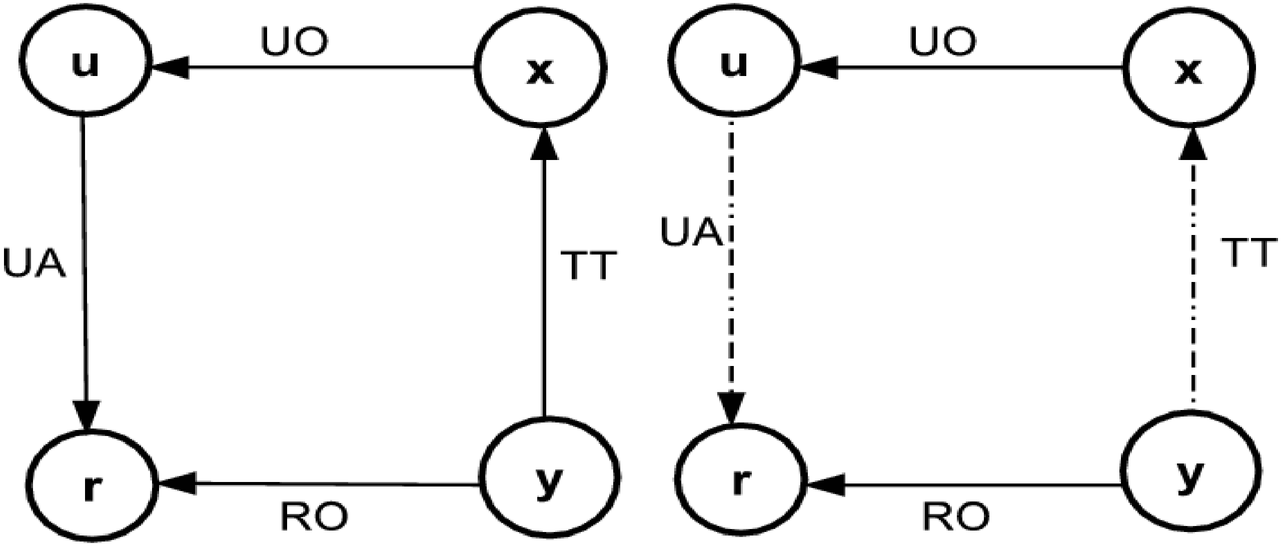


Figure 6: An Example of Edge Attributes in Relationship Graph Expressible in ReBAC$_{BE}$

Example (Continued…)



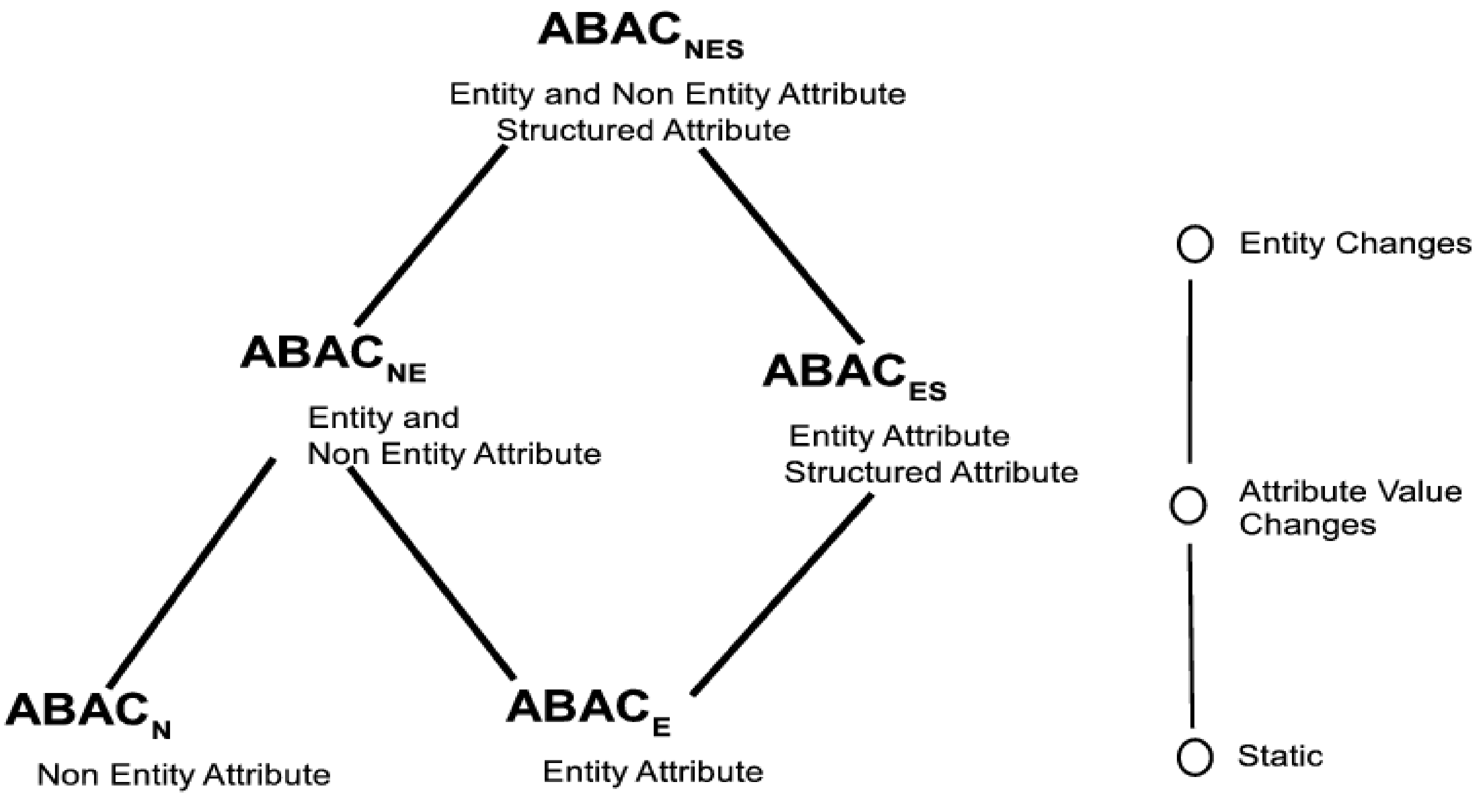
Structure Edge Attribute:
dependsOn

Sub Attributes of dependsON
Source Node
Target Node
RelationshipType

dependsOn (u,r,UA) = (y,x,TT)

Figure 7: An Example of Node Attributes in Relationship Graph Expressible in ReBAC$_{BNES}$ [Cheng et al. 2016]

ABAC Classification

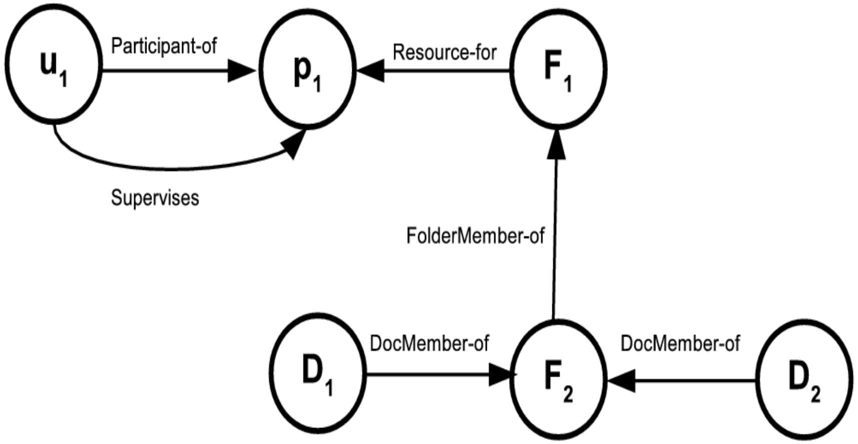


(a) ABAC Structural Models

(b) ABAC Dynamics

Figure 8: ABAC Framework

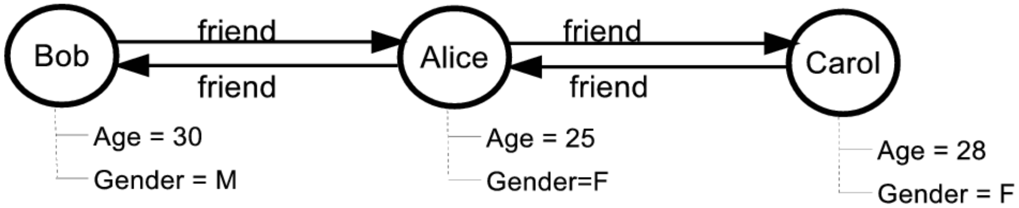
Expressing Relationship Graph with Attributes



- Entity types = {user, project, folder , document}
- Attributes:
  - ❑ User attributes ={Participant-of, Supervises}
  - ❑ Folder attributes = {Resource-for, FolderMember-of}
  - ❑ Project attributes = {}
  - ❑ Document attributes ={DocMember-of}

Relationship Graph in Figure 4 is Expressible with ABAC_E

Expressing Relationship Graph with Attributes (Continued…)

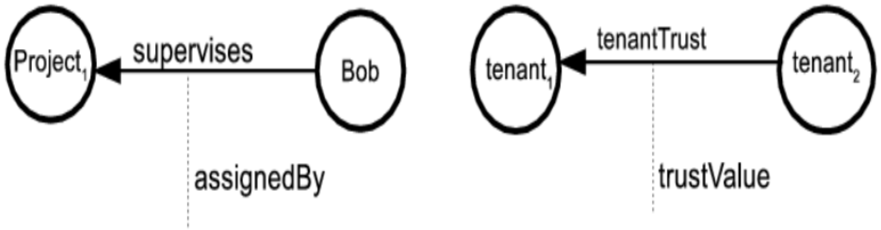


entityType = {user}

Attribute:

- ❑ User’s entity attribute ={friend}
- ❑ User’s Non Entity Attribute ={Name, Age, Gender}

Relationship Graph in Figure 5 is Expressible with ABAC_{E}

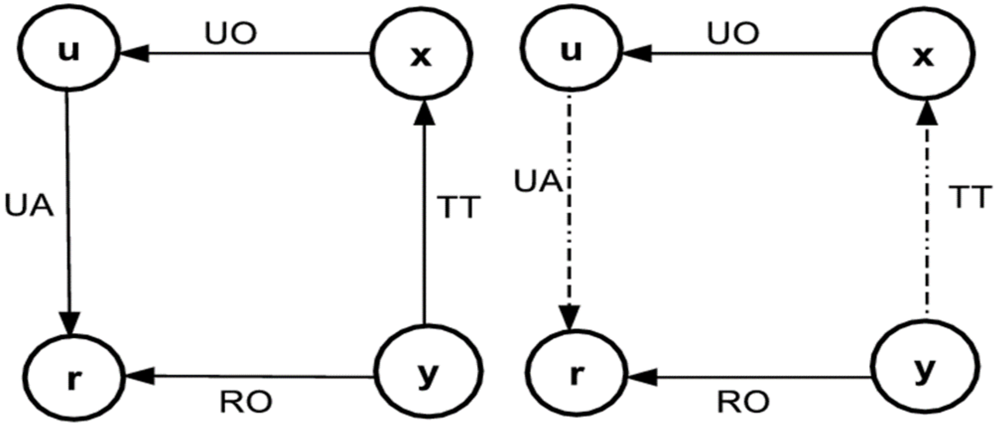


- entityType = {user, project, tenant}
- Attribute:
  - ❑ user’s atomic entity attribute ={supervises}
  - ❑ User’s structured entity Attribute ={assignedBy}

e.g. assignedBy(Bob) = (“Project1”, “supervises”, “Alice”)

Relationship Graph in Figure 6 is Expressible with ABAC_{ES}

Expressing Relationship Graph with Attributes (Continued…)



- Entity types: {user, tenant, role}
- Attribute:
  - ❑ User’s atomic entity attribute: {UO,UA}
  - ❑ Users Structured Entity Attribute: {dependentEdge}

  dependentEdge(u) = (“r”,“UA”, {(y,x,TT)})

Relationship Graph in Figure 7 is Expressible with ABAC_{ES}

Expressing Multilevel Relationship With Attributes

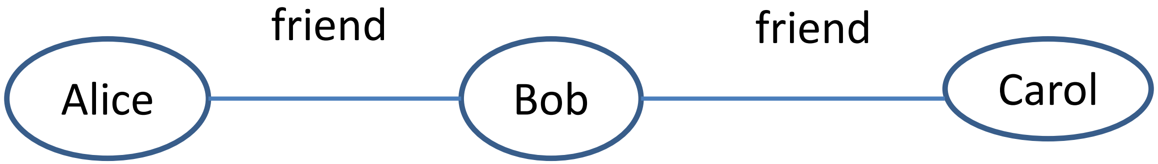


Figure 9. A simple Relationship Graph

Attribute Composition

- ❑ Needs one attribute: friend
- ❑ Policy Expression uses Attribute composition

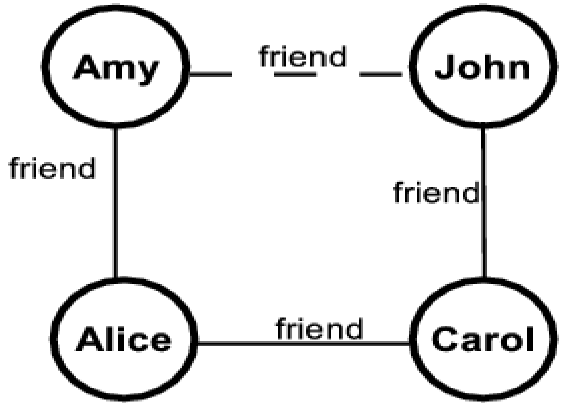
friend(Alice)={Bob}
friend(friend(Alice))={Carol}

Composite Attribute

- ❑ Needs two attribute
  1. friend
  2. friendOfFriend
- ❑ Policy Expression uses direct attributes

friend(Alice) ={Bob}
friendOfFriend(Alice)={Carol}

Example



friend(Alice) = {Amy, Carol}
friendOfFriend(Alice) = {John}

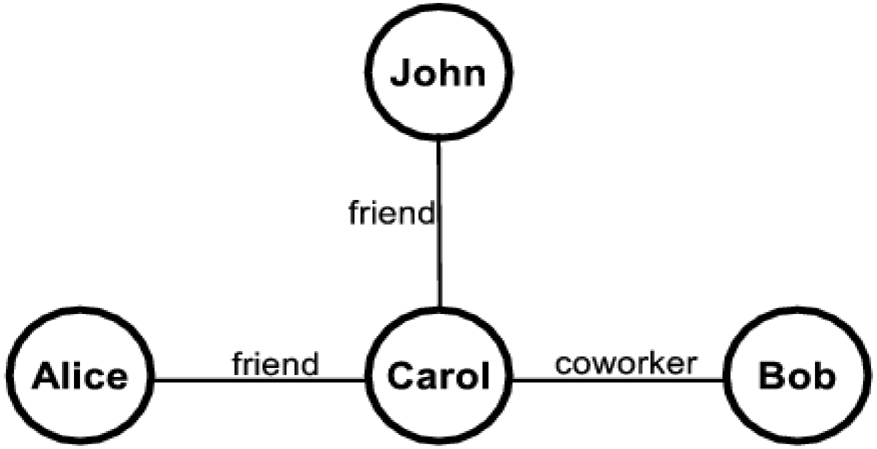
Figure 10. A simple Relationship Graph

If the friend relationship between Amy and John deleted

friendOfFriend(Alice) = ?

Instead of keeping the end user as attribute value we have to keep the exact path information.

Example



Attribute Composition:

friend (“Alice”) = {“Carol”}
coworker (“Alice”) = { }
friend (friend(“Alice”)) = { “John”}
coworker(coworker(“Alice”)) = { }
friend (coworker(“Alice”)) = { }
coworker (friend(“Alice”)) = {“Bob”}

Composite Attribute:

friend (“Alice”) = {“Carol”}
coworker (“Alice”) = { }
friendOfFriend(“Alice”) = { “Carol.John”}
coworkerOfCoworker(“Alice”)) = { }
friendOfCoworker(“Alice”) = { }
coworkerOfFriend(“Alice”)) = {“Carol.Bob”}

Figure 12: Multilevel Relationship Expression with Attribute

Comparison: On Dynamics

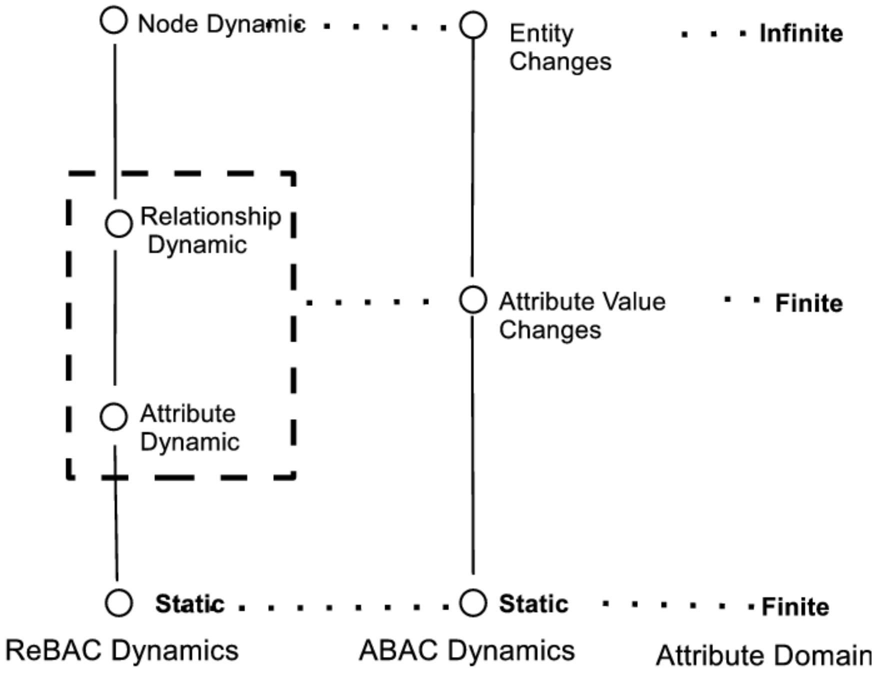


Figure 12: ReBAC Dynamics, ABAC Dynamics and Attribute Domain wise Comparison between ReBAC and ABAC

$ABAC_X \equiv ReBAC_Y \; Means$

- Static and finite attribute domain $ABAC_X \equiv Static \; ReBAC_Y$
- $ABAC_X \; Attribute \; value \; changes \; with \; finite \; domain \equiv Relationship \; Dynamic \; ReBAC_Y$
- $ABAC_X \; with \; entity \; changes \; and \; infinite \; domin \; entity \; attribute \equiv node \; dynamic \; ReBAC_Y$

Comparison: Equivalent Structural Models for ReBAC and ABAC

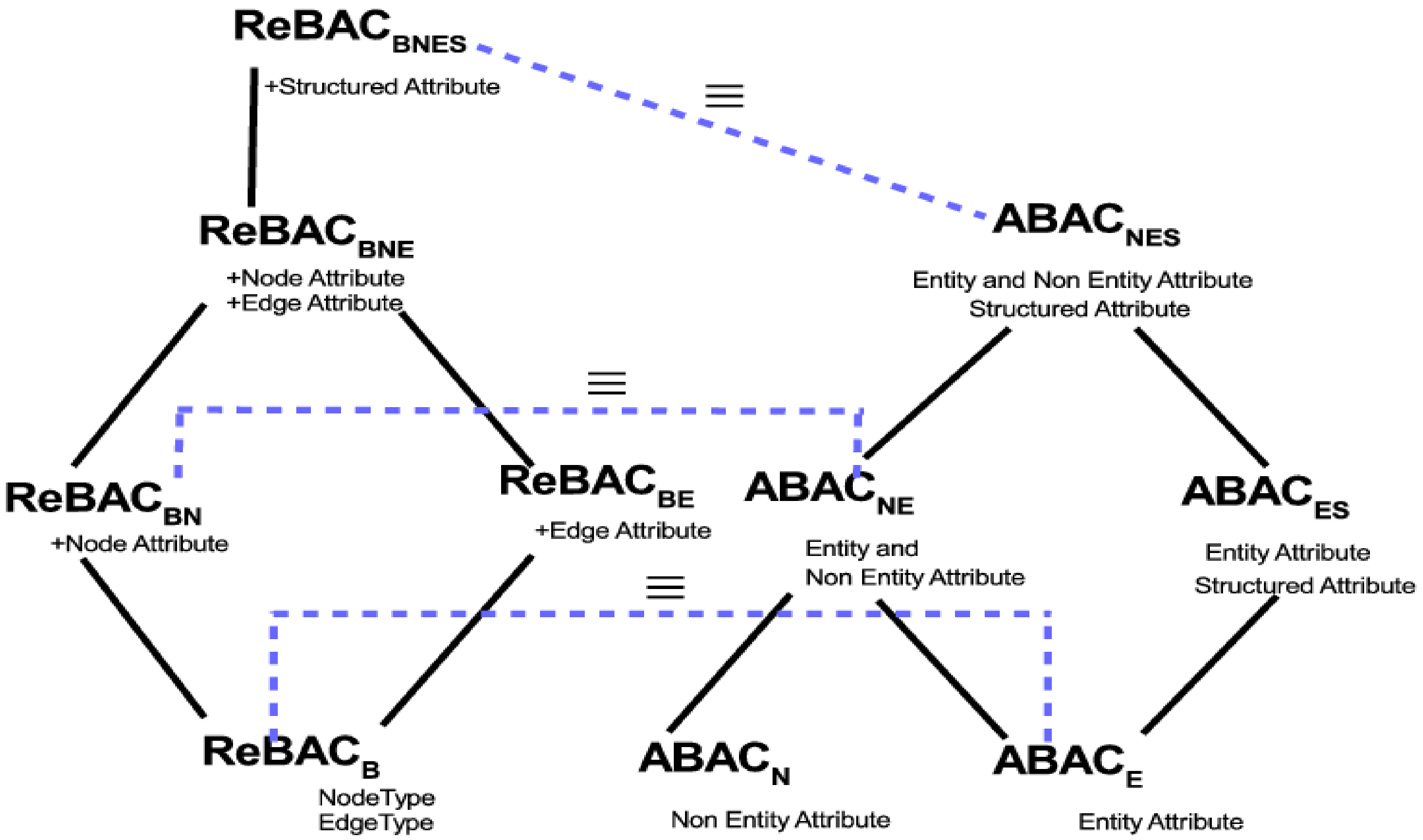


Figure 13: Equivalence of ReBAC and ABAC Structural Classification

Comparison: Non-Equivalent Structural models for ReBAC and ABAC

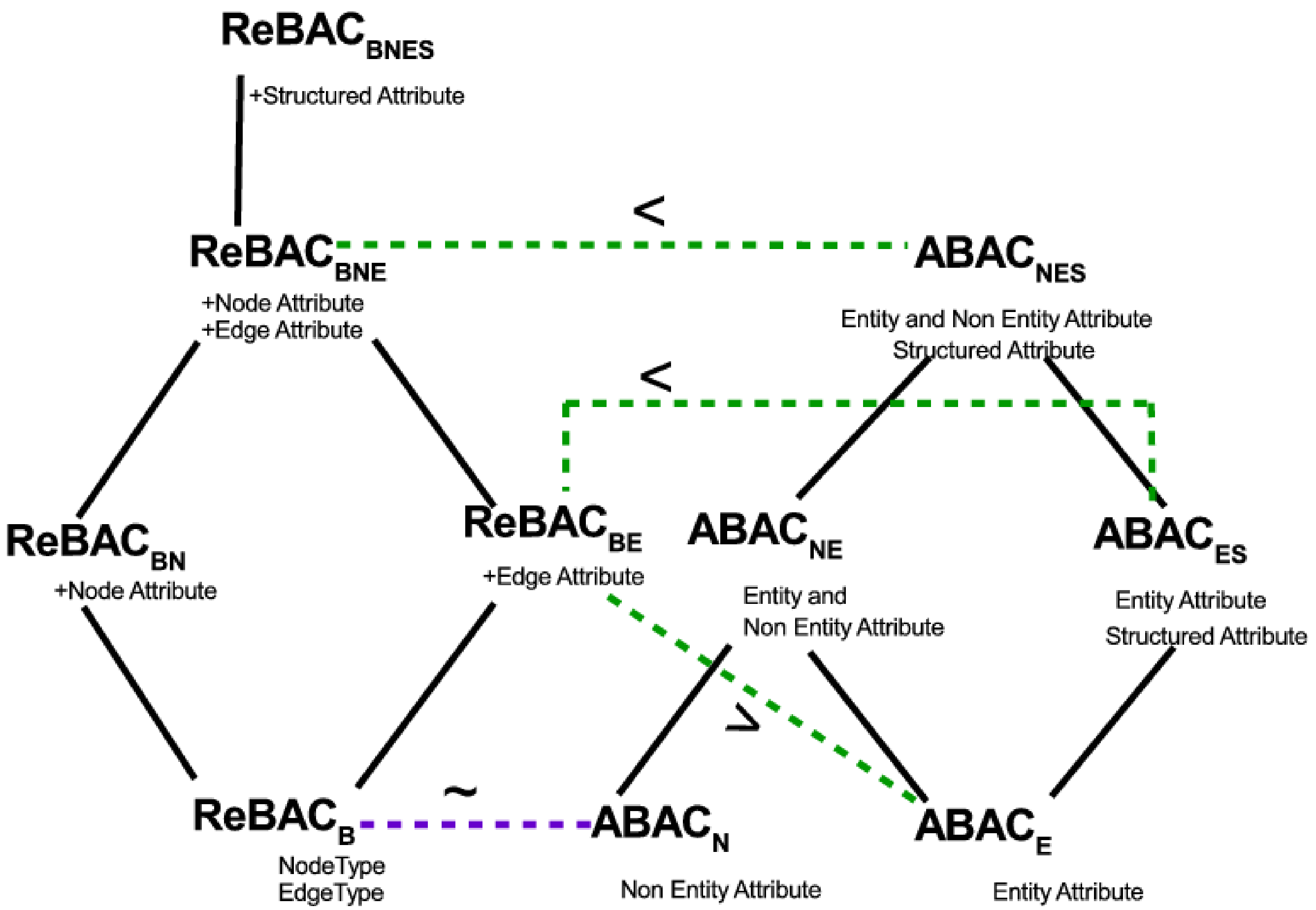


Figure 14: Non-Equivalence of ReBAC and ABAC Structural Classification

Comparison: On Performance

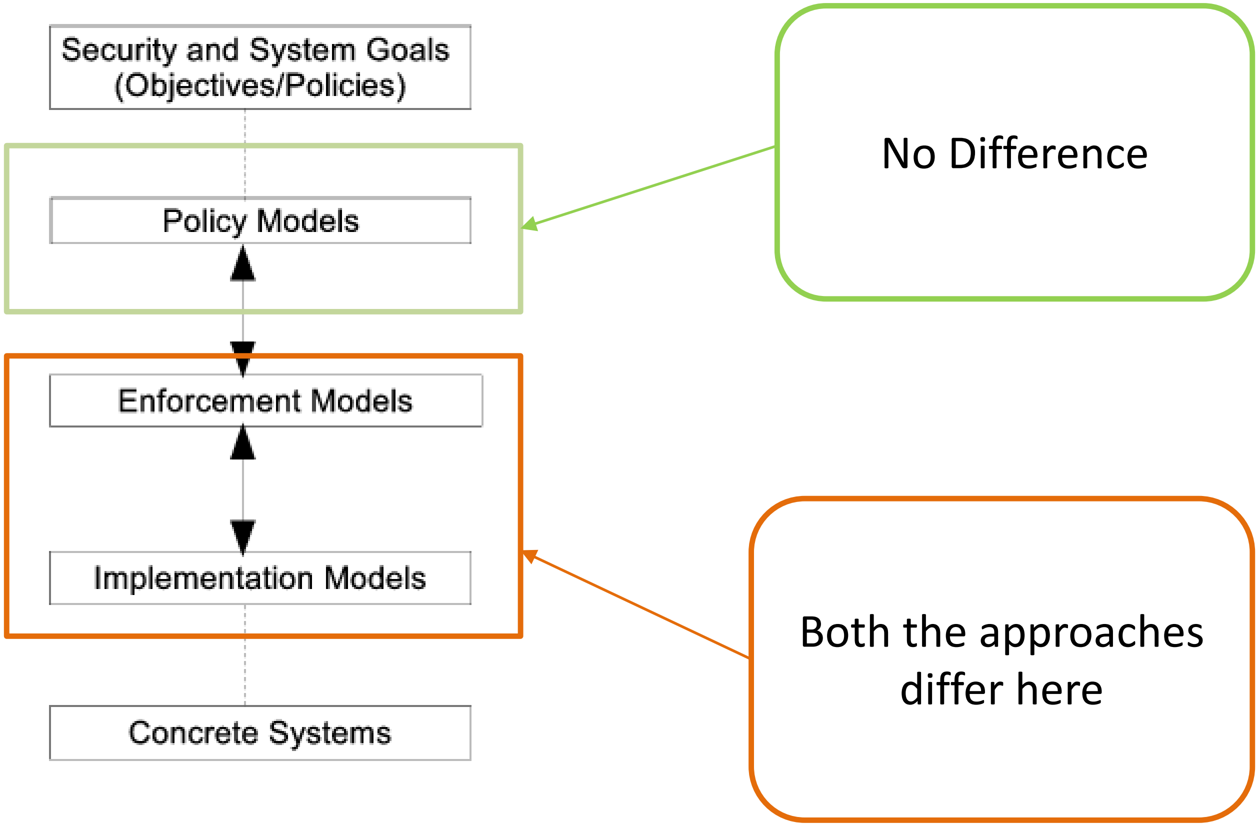
- Attribute Composition is similar to ReBAC and Both have polynomial complexity for authorization policy and constant complexity on update
- Composite attribute has constant complexity on authorization policy and polynomial complexity on update to maintain relationship changes.
- Performance Depends on :
  - ❑ Node Dynamics
  - ❑ Relationship Dynamics
  - ❑ Density of the Relationship Graph

Comparison: Choice of Models

- For static system or only non entity attribute change------Composite attribute is the best approach
- System with huge node dynamics, relationship dynamics and high relationship density----- Attribute composition is the best option
- If the system is in the middle between two extremes ---- A hybrid approach where both composite attribute and attribute composition is used.
- Hybrid Approach:

To achieve p level relationship composition it uses m level composite attribute and n level attribute composition where p = n X m.

Comparison: In Respect of PEI Framework



No Difference

Both the approaches differ here

Figure 15: PEI Framework

Conclusion

- Our results indicate that the relationship between ABAC and ReBAC is subtle and variable depending on the precise flavor of these two access control approaches in any given model. At the same time we are able to make some general statements about this comparison.

- Metrics beyond theoretical equivalence need to be brought into consideration to better understand the relative advantages and disadvantages of these two approaches. Performance is one such metrics but others such as maintainability, robustness, and agility, also need to be studied.

Questions/Comments