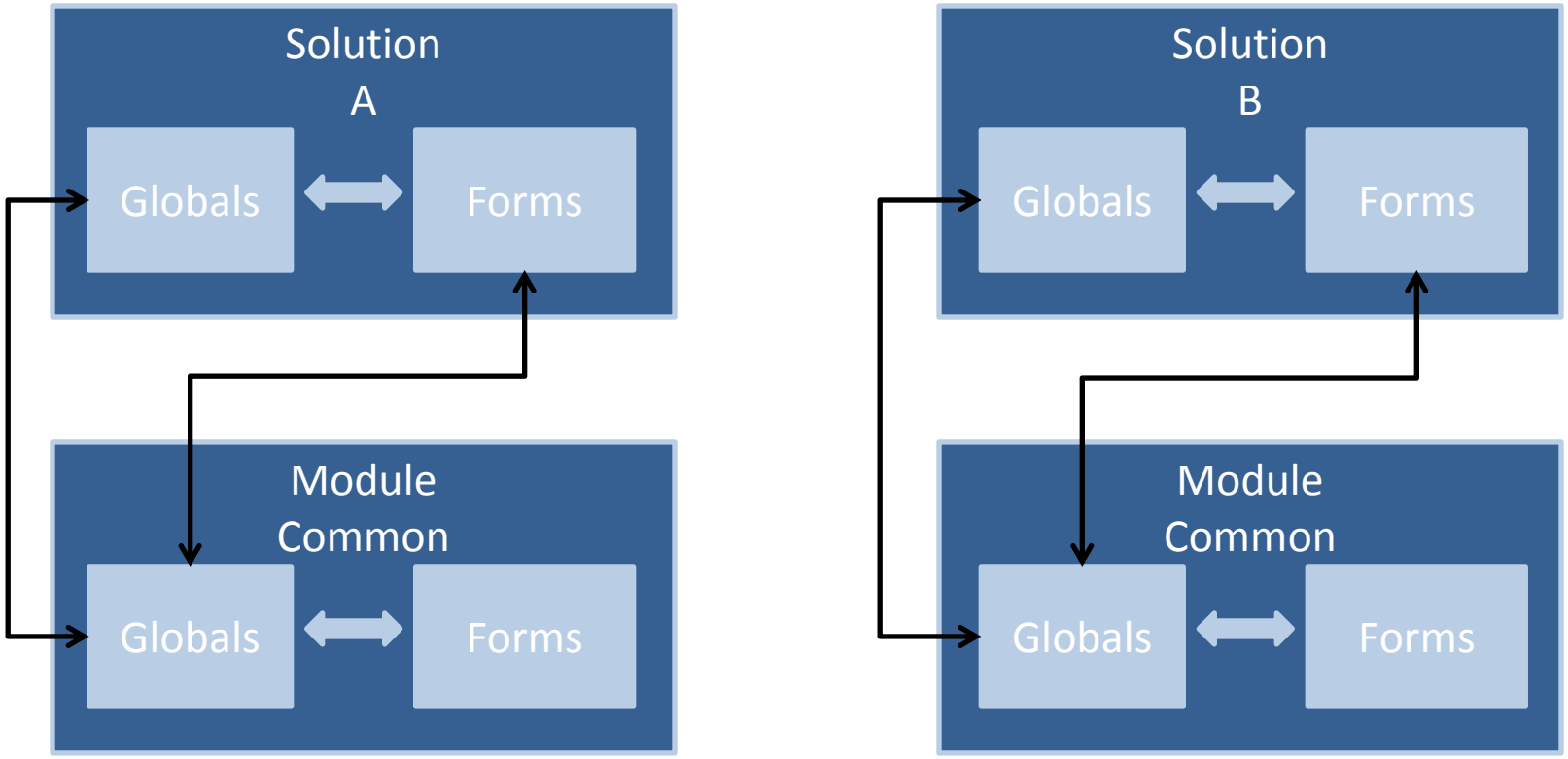
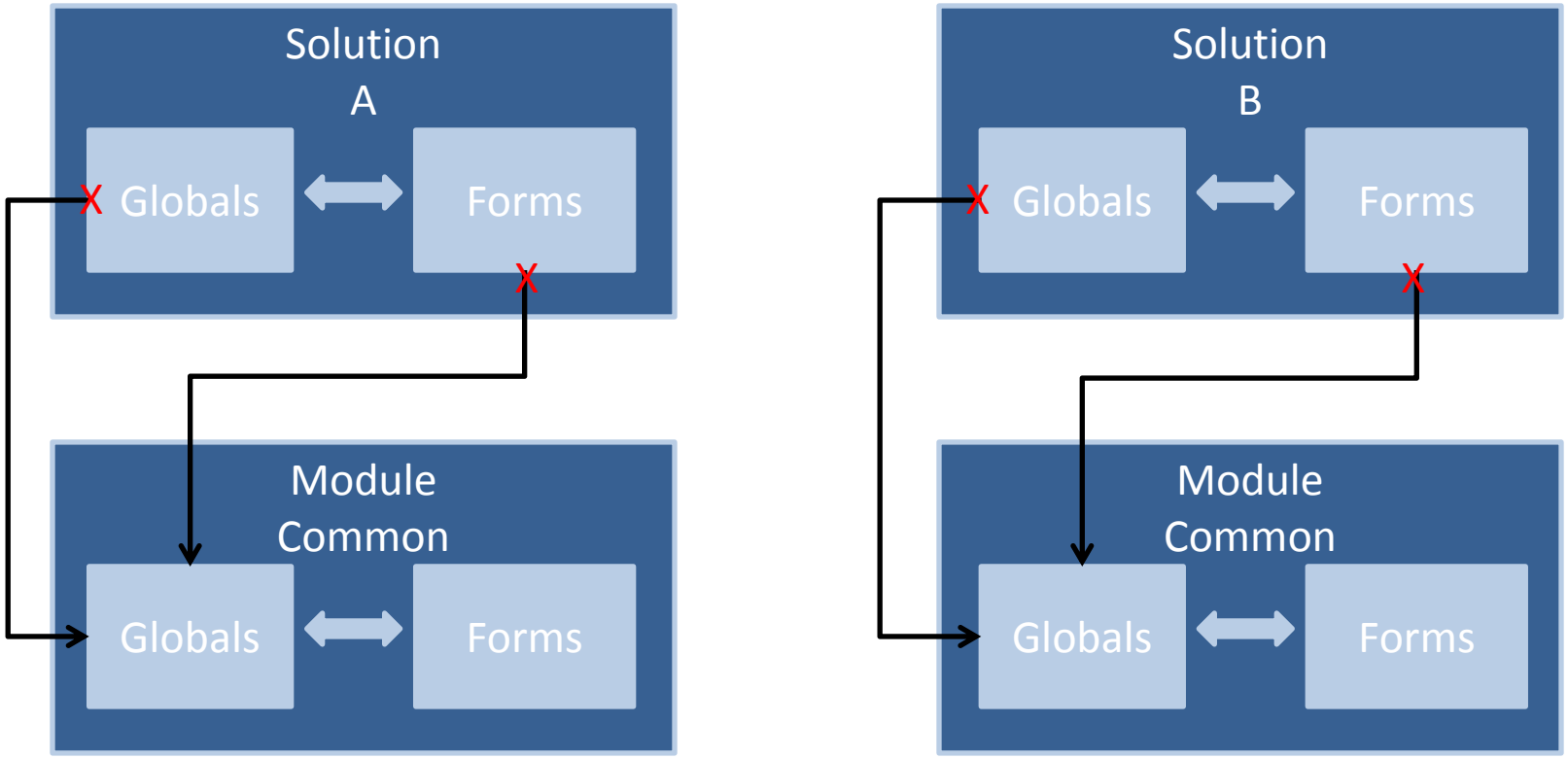
Solution and Module Scope for v3.x through 5.x



Module Common containing solution independent methods like AddNew, Edit, hideComboBoxes, etc. is shared among numerous solutions. Its ability to reference application wide business logic or form specific business logic has been its key to its universal deployment among multiple solutions.

A form or a form method in Solution A calls a globals method in module Common like AddNew or Edit. The method will call other methods in module Common to prep the form for user entry, prevent the user from navigating away from the form, etc. This same method will also call solution A specific global methods and form specific methods based on the set of either or both application and form specific rules.

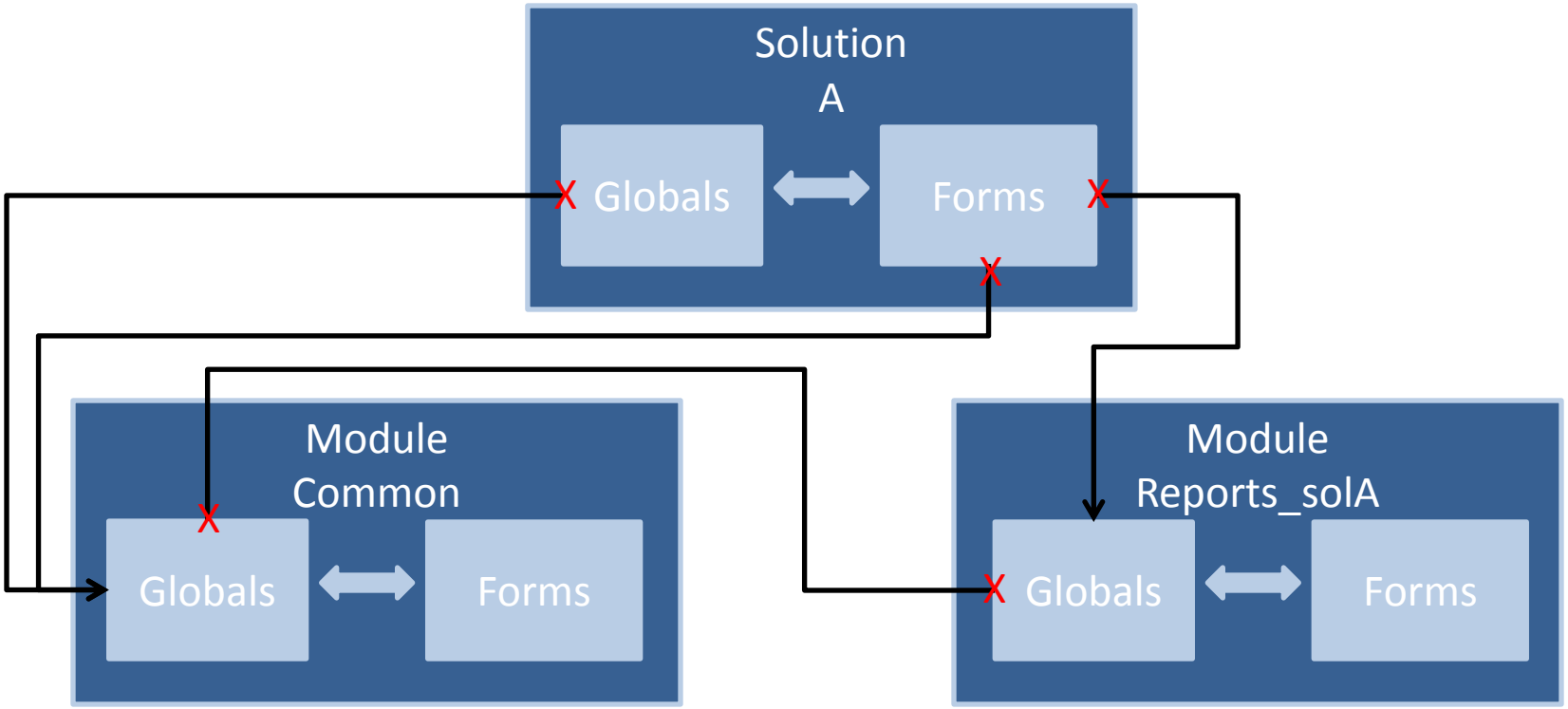
Solution and Module Scope for 6.x



Module Common containing solution independent methods like AddNew, Edit, hideComboBoxes, etc. is shared among numerous solutions. Its ability to reference application wide business logic or form specific business logic has been its key to its universal deployment among multiple solutions.

A form or a form method in Solution A calls a globals method in module Common like AddNew or Edit. The method will call other methods in module Common to prep the form for user entry, prevent the user from navigating away from the form, etc. This same method will also call solution A specific global methods and form specific methods based on the set of either or both application and form specific rules.

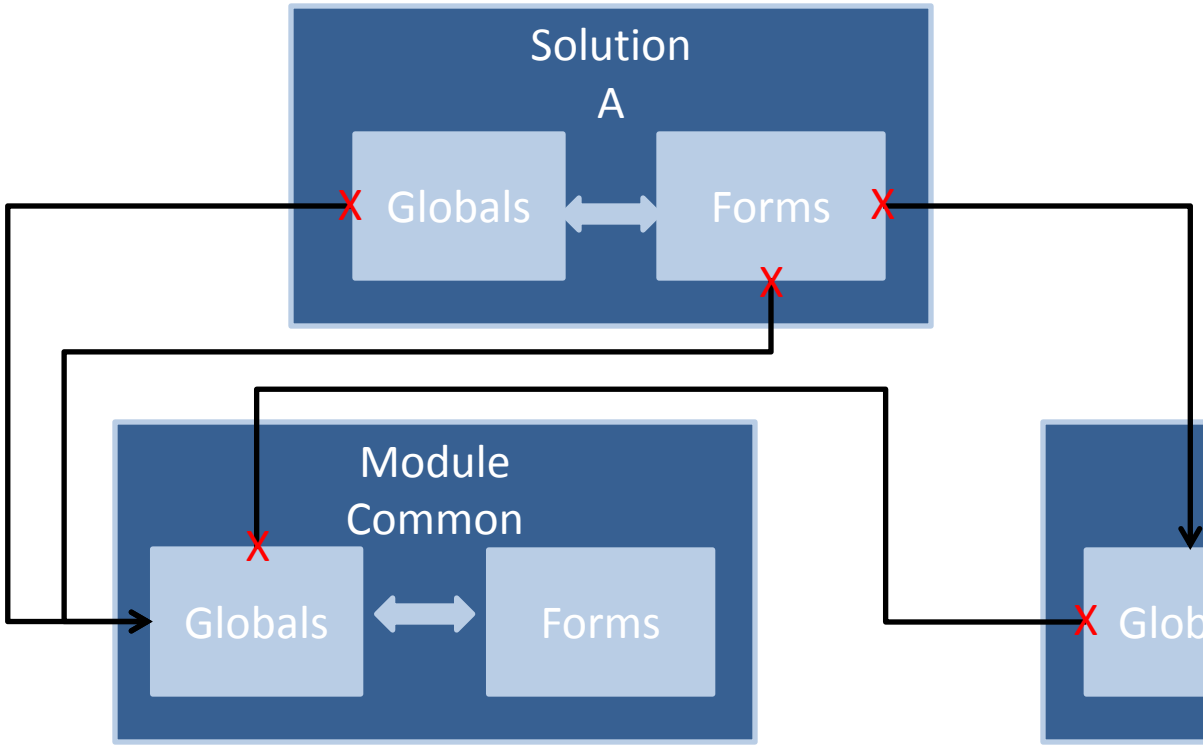
Solution and Module Scope for 6.x



The ability to reference application wide business logic or form specific business logic has been the key to the universal deployment of modules among multiple solutions.

The ability of a global method to call solution A specific global methods and form specific methods based on the set of either or both application and form specific rules is needed. The analogy here is that the parent can see the child's but the child cannot see the parent's attributes and behaviors. The global methods are the behaviors and the global variables are the attributes of the either the parent or child object. Here in 6.x, the child can neither see the parent nor a sibling, in effect creating alternate definitions of the term global, i.e. solution global, module A global, module B global, et cetera.

Proposed Solution and Module Scope for 6.x Fix



The ability to reference application wide business logic or form specific business logic has been the key to the universal deployment of modules among multiple solutions.

Proposal One: A 6.x solution level property that when set true allows the behavior as in Servoy v3.x through Servoy 5.x. The property could default to false.
or **Proposal Two**: Two 6.x solution level properties, the first solution level property would affect the child to parent relationship so that when set to true, the child would be allowed to see the global methods and global variables declared in the parent. A second solution level property would affect the child to sibling relationships so that when set to true, the global methods and global variables declared in a sibling module would be visible to another sibling module and vice-versa. Both of these solution level properties could default to false.
For completeness a third solution level property when set to true, would allow any child (such a grandchild or great-grandchild, etc.) to see the global methods and global variables declared by any of its ancestors. This third solution level property along with the first two solution level properties would be the functional equivalent of the single solution level property describe in Proposal One. The defaults could be false.