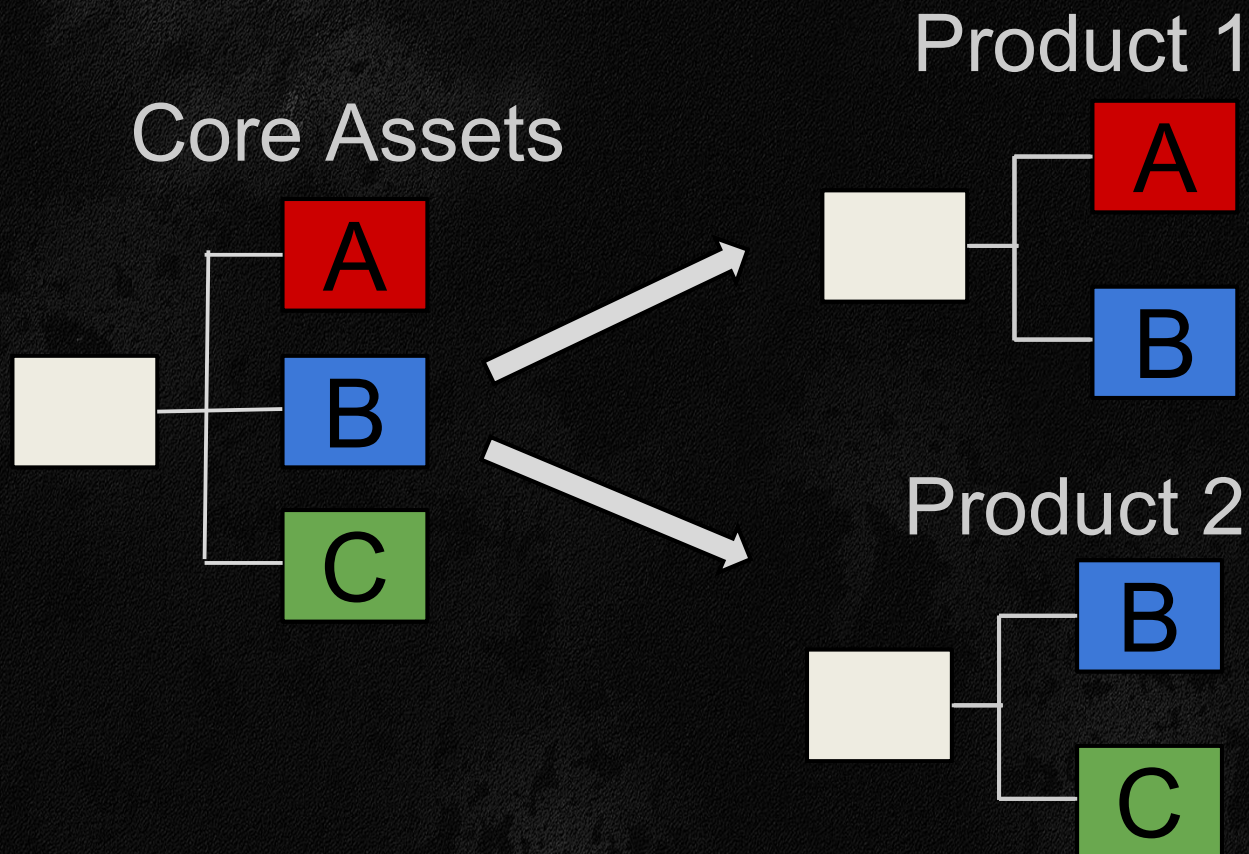


# Managing the Evolution of Software Product Lines

Cheng Thao & Michael Haufe {chengt,mlhaufe}@uwm.edu

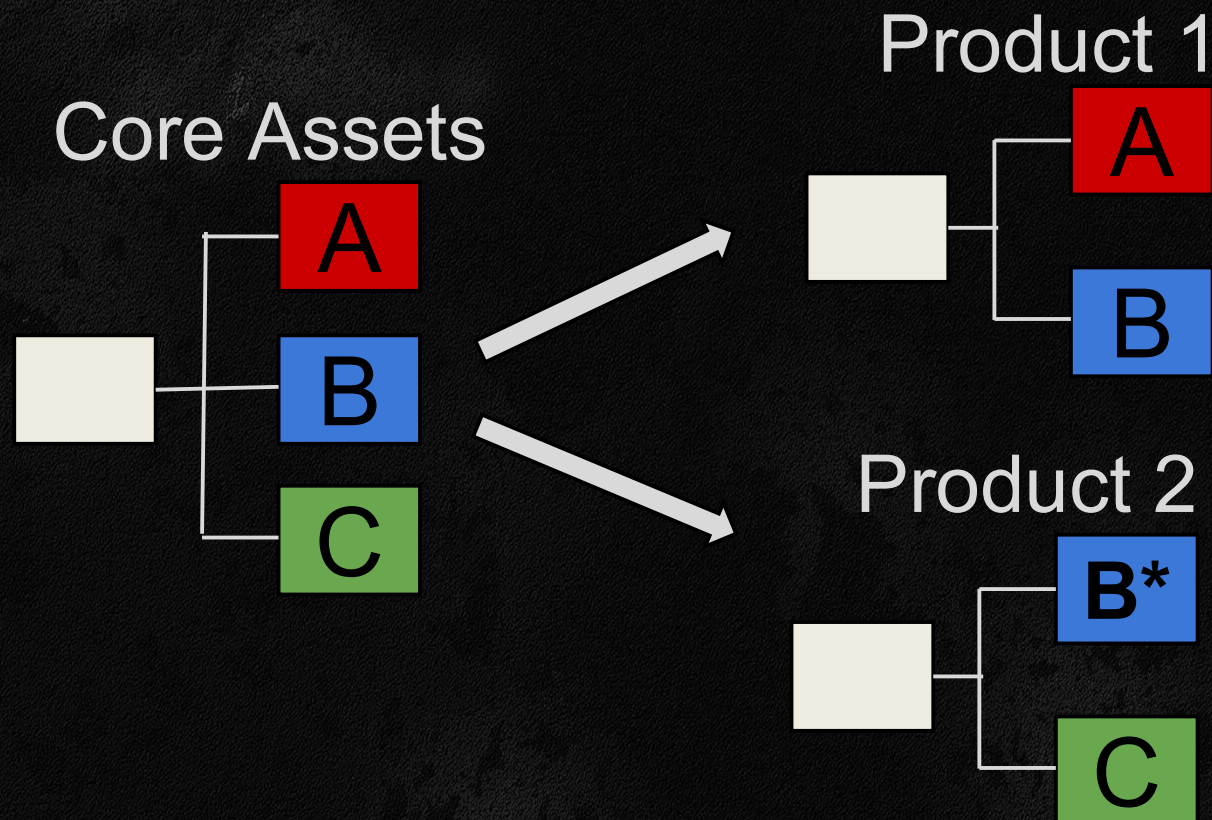


- Software Product Line Engineering (SPLE) is a systematic approach to software reuse
- Developers produce a set of core assets that are intended to be shared by multiple products





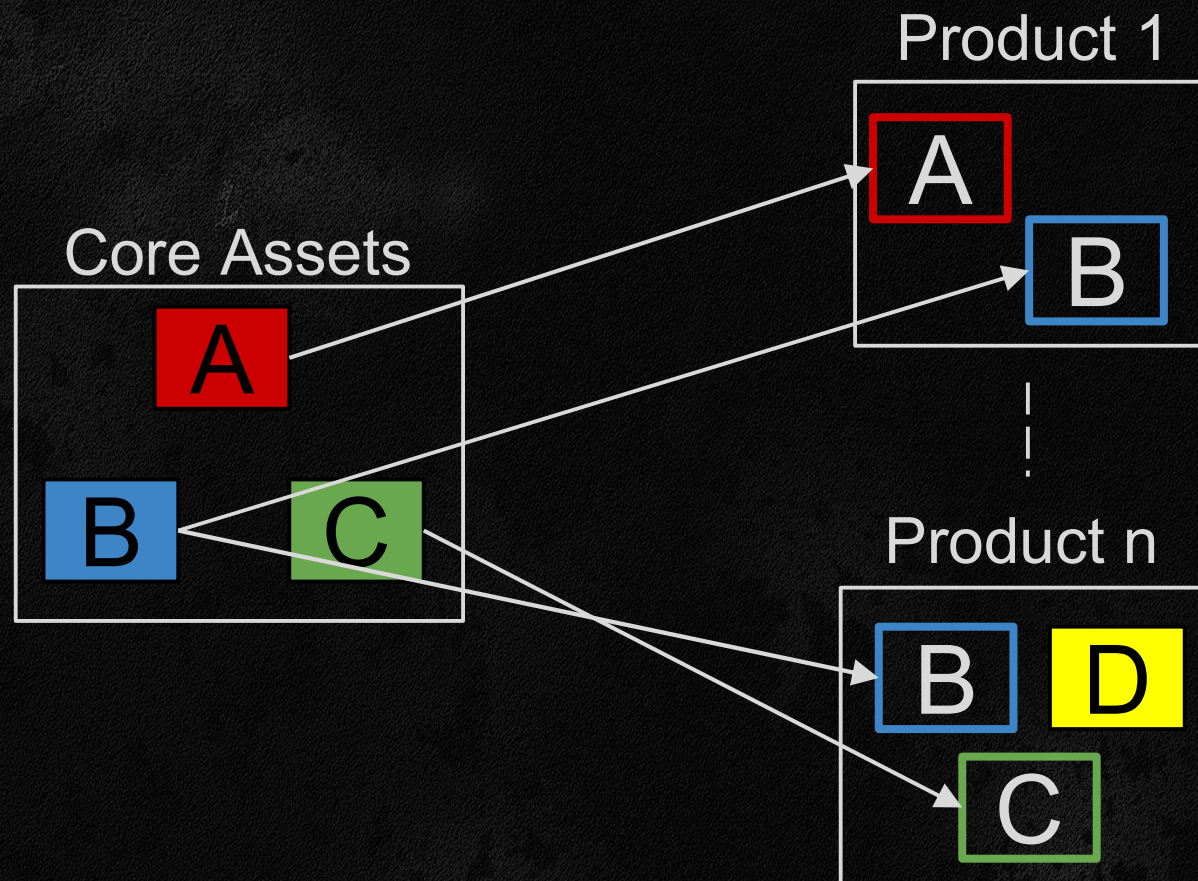
- Each of the products has some of its own product specific code
- All of this code can evolve over time for the normal reasons of maintenance and evolving functionality.





Reuse is a central theme of product line engineering

Our approach uses shared components to enable reuse.





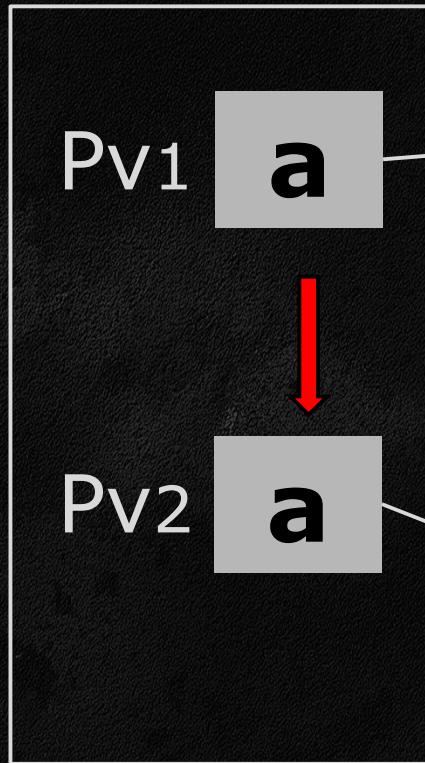
Managing the evolution of a product line is challenging because of the independent evolution of products, core assets, and their interactions

We think of product line evolution as a two dimensional problem where one dimension is time (artifacts changing over time) and the other is space (artifacts varying between products).

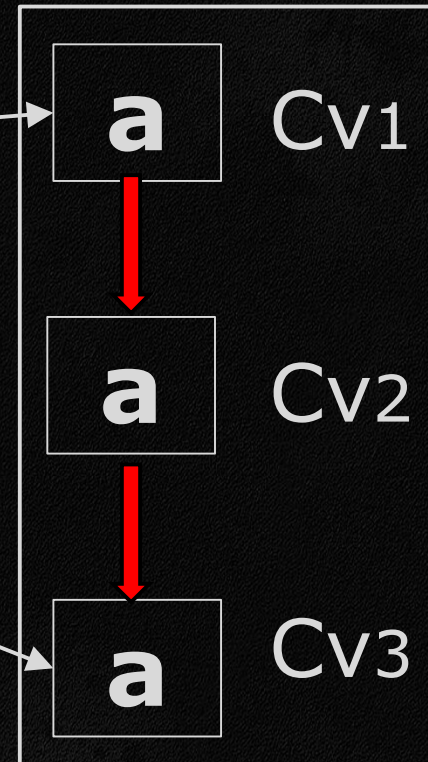
Evolution



## Product Version Space




## Core Assets Version Space



Shares

Shares

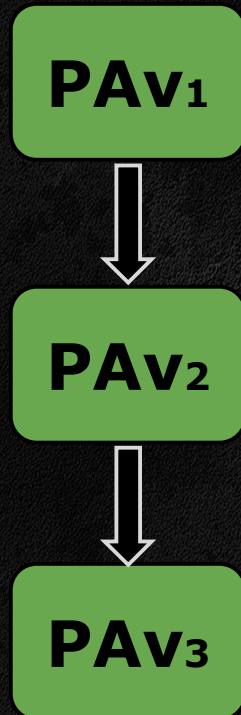
 Shared  
component

 Core asset

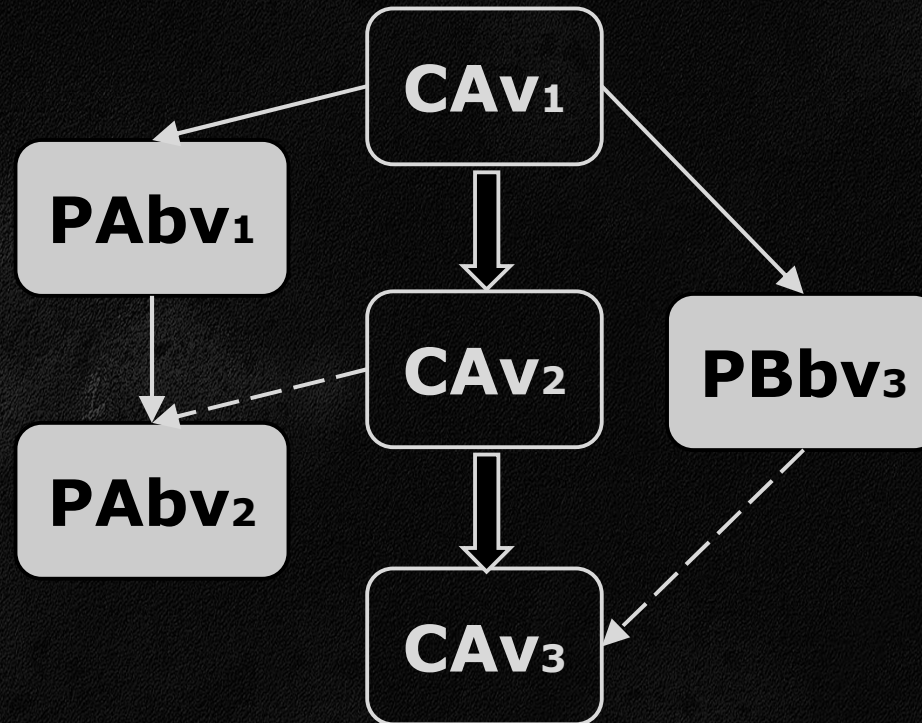
In the example above, the product is using shared component "a" from the core assets. However, as the product moves from version "Pv1" to version "Pv2", the version of the shared component changes from "Cv1" to "Cv3"



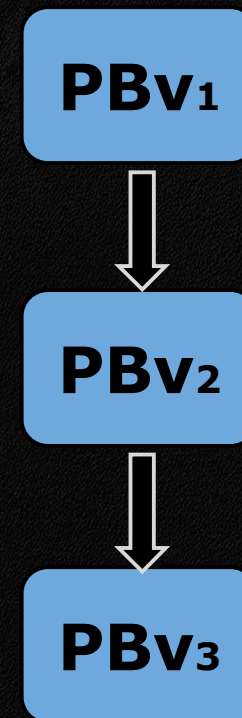
### Product A Version Tree



### Core Assets Version Tree



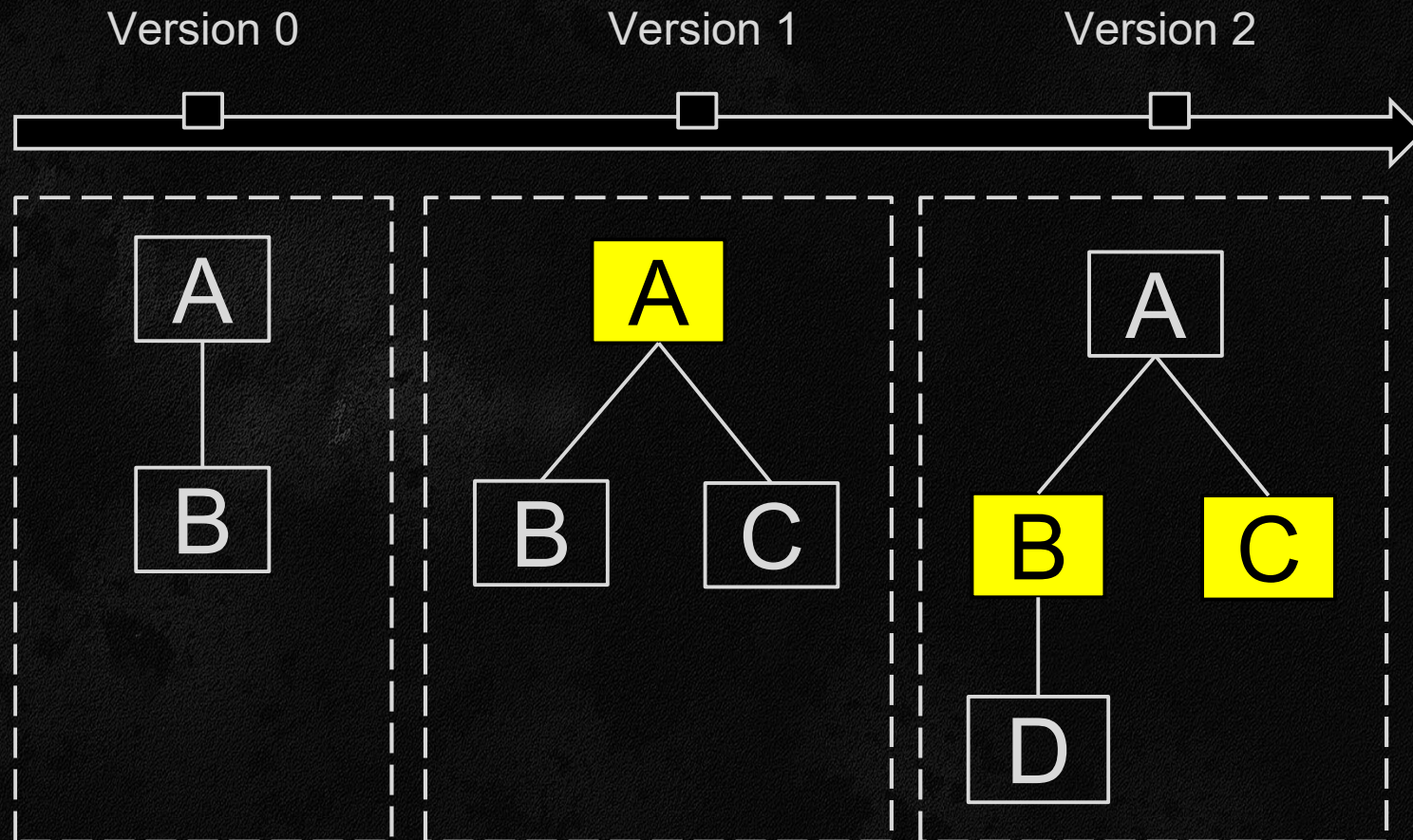
### Product B Version Tree



The larger black arrows indicate the main development trunk while the smaller arrows indicate branches. The dotted lines indicate a merge.



# Product Versioning Model



A version represents a state of the project tree and its artifacts. Each box represents a file or directory. A colored box indicates a file or directory that has changed since the previous version.



# Evaluation



- We evaluate our prototype using the Graph Product Line (GPL)
- Since graphs are well understood data structures, they are a prime candidate for representing an abstraction of a Product Line
- Our goals are to determine if our prototype can model a realistic software product line, capture its evolution, and evaluate change propagation between core assets and products.



GUI screenshot of a GPL program option screen which provides a means of specifying the implementation of a variety of GPL instances.

| Direction                                 | Weight                                    | Search                               | Algorithms                                    |
|---|---|--------------------------------------|---|
| <input checked="" type="radio"/> Directed | <input checked="" type="radio"/> Weighted | <input type="radio"/> DFS            | <input checked="" type="checkbox"/> Number    |
| <input type="radio"/> Undirected          | <input type="radio"/> Unweighted          | <input checked="" type="radio"/> BFS | <input type="checkbox"/> Connected Comp.      |
|   |   |                                      | <input type="checkbox"/> Strongly Con. Comp.  |
|   |   |                                      | <input type="checkbox"/> Cycle Checking       |
|   |   |                                      | <input type="checkbox"/> MST Prim             |
|   |   |                                      | <input type="checkbox"/> MST Kruskal          |
|   |   |                                      | <input type="checkbox"/> Single Shortest Path |



# Results



We evaluate our prototype with the GPL to confirm different change propagation cases. The prototype supports all the cases listed in the table below. Bold items are concrete components, while items in non-bold characters are shared components. The ' and \* represent different changes.

|      | Before    |          | After      |         |
|------|-----------|----------|------------|---------|
| Case | Core      | Product  | Core       | Product |
| 1    | <b>a'</b> | a        | <b>a'</b>  | a'      |
| 2    | <b>a'</b> | a*       | <b>a'</b>  | a*'     |
| 3    | <b>a'</b> | a*       | <b>a'</b>  | a'      |
| 4    | <b>a</b>  |          | <b>a</b>   | a       |
| 5    | <b>a</b>  | a*       | <b>a*</b>  | a*      |
| 6    | <b>a'</b> | a*       | <b>a'*</b> | a*      |
| 7    | <b>a'</b> | a*       | <b>a*</b>  | a*      |
| 8    |           | <b>a</b> | <b>a</b>   | a       |



# CONCLUSION

We have described an approach that is capable of versioning multiple types of product line projects including Graph Product Lines. It has a versioning model for a product line consisting of a single core assets project and multiple product projects where core assets are shared among the products through the use of shared components. Using the shared component data structure and the branching of the core assets project, we are able to support independent development of core assets and products and change propagation between them.