

# Strategic reuse with software product lines

Paulo Borba  
Informatics Center  
Federal University of Pernambuco

[phmb@cin.ufpe.br](mailto:phmb@cin.ufpe.br) ♦ [twitter.com/pauloborba](https://twitter.com/pauloborba)

Problems with ad hoc  
approach

Software product lines basic  
concepts

Problems  
with used  
approach?

# Problems

- Use case and code duplication
- Reuse is achieved via property files only
- Badly structured, difficult to maintain, use cases and code
- Products organized as branches
- No high-level automatic product generation
- No high-level visualization of products and features

# Problems due to focus on product...



[www.usm.maine.edu](http://www.usm.maine.edu)

# not product family!



it.zaobao.com

Products with common functionality,  
but with variations too

# From hardware products to software products...

Program families are sets of programs whose common properties are so extensive that it is advantageous to study the common properties of the programs before analyzing individual members

Adapted from: On the Design and Development of Program Families (Parnas 1976)

# Different devices, 15 to 60 different applications...



Different clients,  
different products

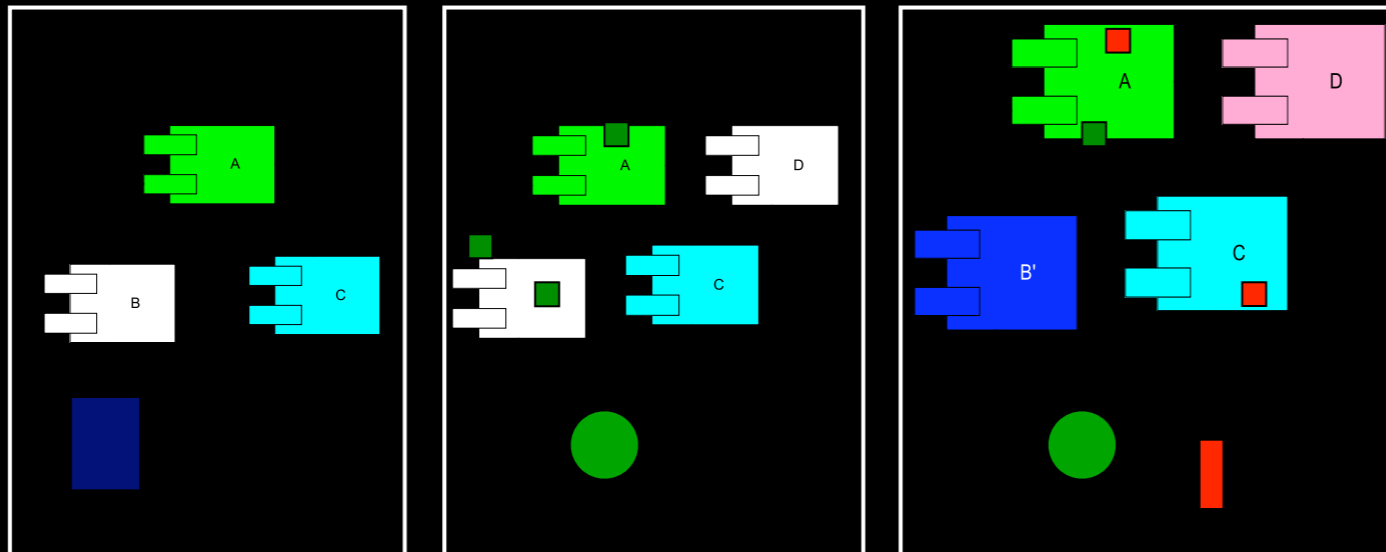
64kb, flip

4Mb, flip

100Kb, sem flip

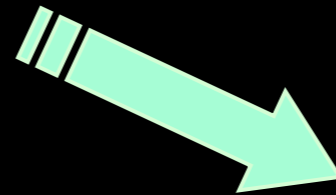


# Little reuse and agility, high costs



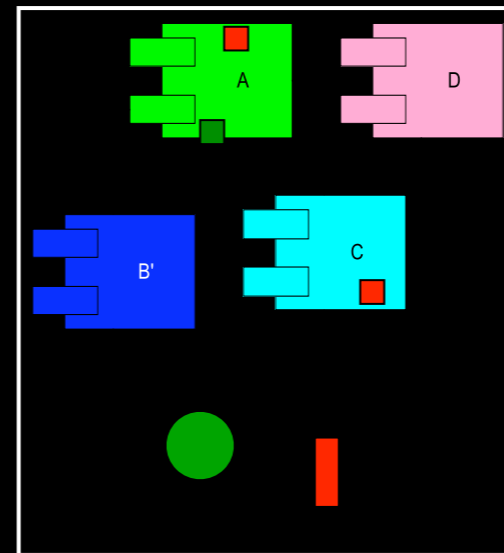
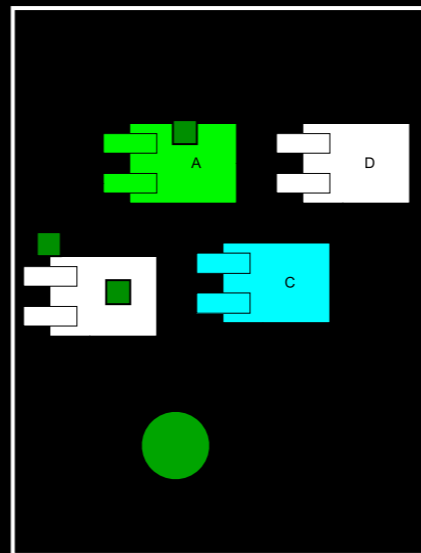
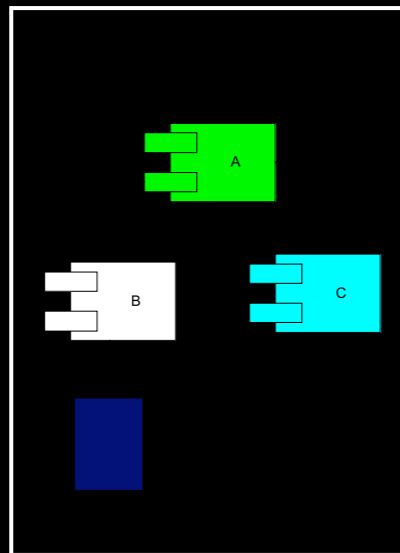
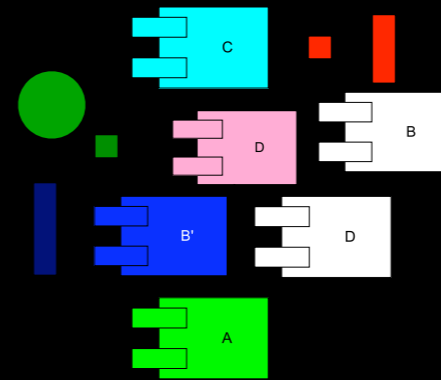
Even with J2ME!

```
WinDiff
File Edit View Expand Options Mark Help
jenemy.java C:\vander\research\projects\cesar\rain_latest\src\org\cesar\rain: C:\vander
90 public void draw(Graphics g) {
91     if(this.isVisible()) {
92         int offsetX = 0;
93         if (this.collisonCount <= 0) {
94             // Draw the dragon
95             g.setClip(this.getX(),
96                 this.getY(),
97                 this.getWidth(),
98                 this.getHeight());
99
100             if (this.getSpeed() > 0) {
101                 if (this.openMouth > 0 || (this.isSpecial && this.isFiring)){
102                     offsetX = -7*(this.getWidth());
103                 } else {
104                     offsetX = -1*(MainCanvas.FrameW*2 + (this.getWidth()));
105                 }
106                 g.drawImage(this.getImage(),
107                     this.getX()+offsetX,
108                     this.getY(),
109                     g.TOP | g.LEFT);
110             } else {
111                 if (this.openMouth > 0 || (this.isSpecial && this.isFiring)){
112                     offsetX = 0;
113                 } else {
114                     offsetX = this.getImage().getWidth()-this.getWidth() - (MainCanvas.FrameW*2 +
115                     this.getWidth());
116                 }
117                 DirectGraphics dg = DirectUtils.getDirectGraphics(g);
118                 dg.drawImage(this.getImage(),
119                     this.getX()+offsetX,
120                     this.getY(),
121                     g.TOP | g.LEFT,
122                     DirectGraphics.FLIP_HORIZONTAL);
123             }
124             // Draw the breath of Fire
125             if (this.isSpecial && this.isFiring) {
126                 offsetX = (MainCanvas.FrameW*2 +
127                     (Resources.dragonFlame.getWidth()/2));
128                 g.setClip(this.getX()+this.getWidth(),
129                     this.getY()+this.getHeight(),
130                     Resources.dragonFlame.getWidth(),
131                     Resources.dragonFlame.getHeight());
132                 g.drawImage(Resources.dragonFlame.getImage(),
133                     this.getX()+offsetX,
134                     this.getY(),
135                     g.TOP | g.LEFT);
136             }
137         }
138     }
139 }
```

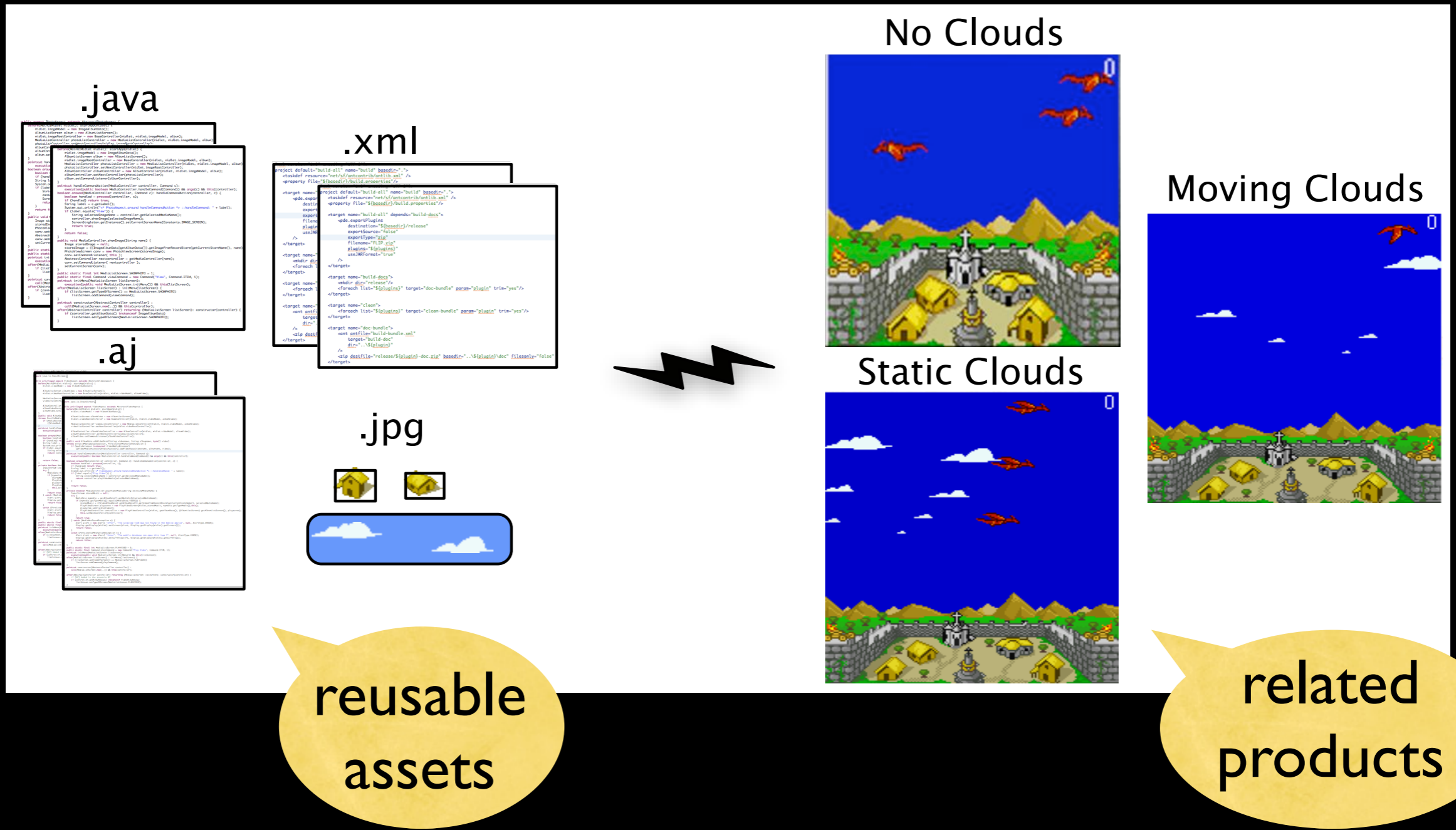


# Solution: product **line**

Strategic reuse of core assets: common artifacts and variations



# Software product line



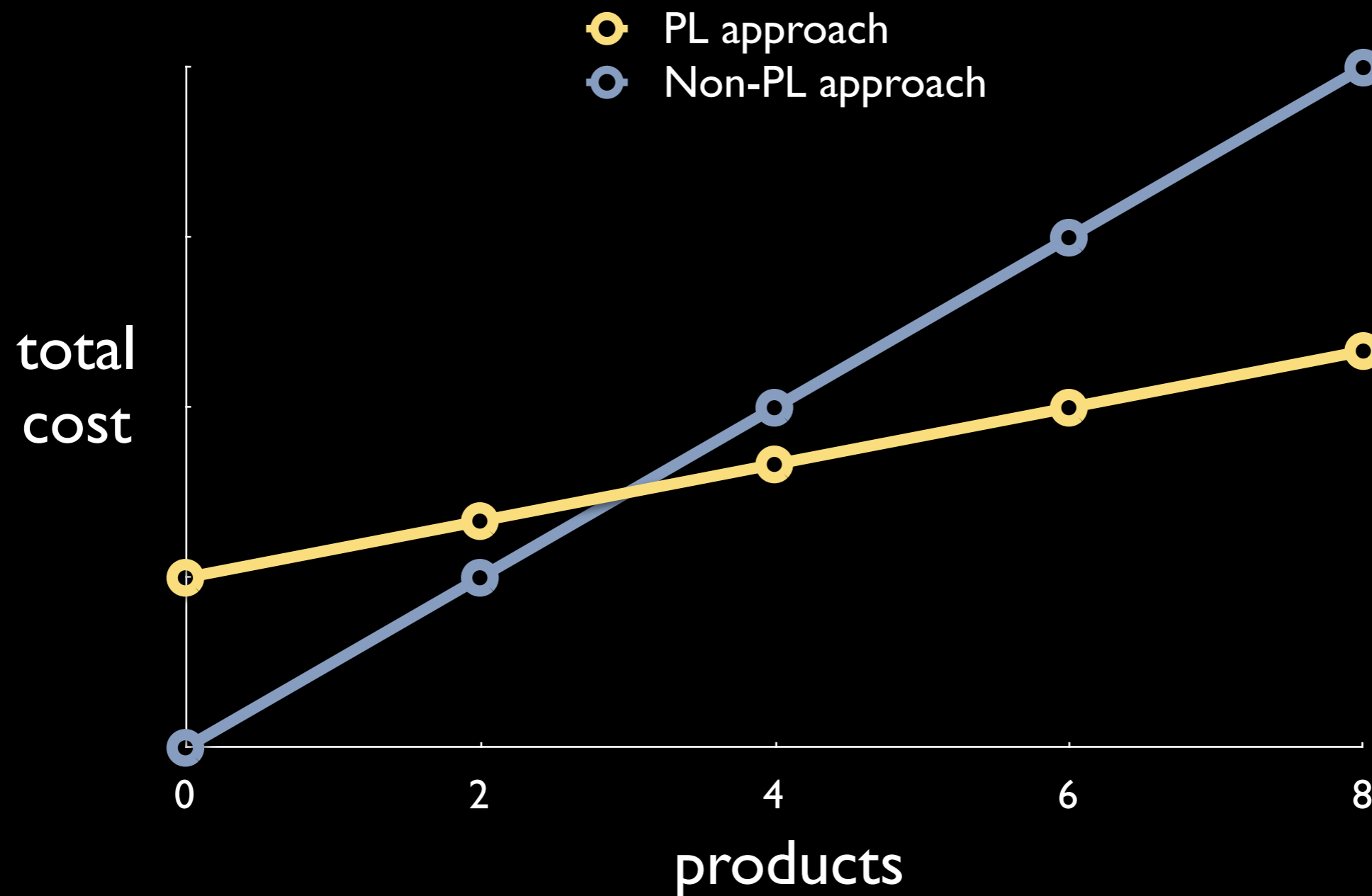
# Family = Line?

- Family
  - Elements
- Line
  - Support that enables the generation of the elements

# PL approach improves...

- customization
- productivity (development and maintenance)
  - 700 builds costing little more than 1
- time to market
- quality

# But not before you have it up and running!



# A software product line is...

a set of software-intensive systems  
sharing a common, managed set of  
features that satisfy the specific needs  
of a particular market segment or  
mission and that are developed from a  
common set of core assets in a  
prescribed way

Software Product Lines, Practices and Patterns.  
Clements and Northrop 2002.

# Feature

- User-visible aspect or characteristic of the family (Kang 1990)
- define both common aspects of the family as well as differences between products in the family
- Logical unit of behavior specified by a set of functional and quality requirements (Bosch 2000)
- Groups requirements



# Yet another reuse approach?

- Text reuse is not reuse!
- Class reuse
  - Failure for business concepts: Client, Account, etc.
  - Even with repositories and search mechanisms
  - OK inside a single project or for infra-structure software
- Component reuse
  - Better than class reuse due to interfaces, deployment and visibility of components
  - Similar problem with business concepts, repositories and search

# Or reuse that pays?

- Services reuse
  - Same as component reuse except for technology involved, and its non technical consequences
- Framework reuse
  - Class or components
  - Successful for both infra-structure (look at GUI) and business (in this case with limited flexibility)
- Knowledge reuse
  - Extremely successful with patterns
- Strategic reuse

Problems with ad hoc  
approach

Software product lines basic  
concepts

# Strategic reuse with software product lines

Paulo Borba  
Informatics Center  
Federal University of Pernambuco

[phmb@cin.ufpe.br](mailto:phmb@cin.ufpe.br) ♦ [twitter.com/pauloborba](https://twitter.com/pauloborba)