Software product lines; ??

Manually fixing your code :

* If you decide to delete a functionality, you can remove the code fragments related to it.

⇒ problem : if a bug occurs, you have to fix it in all functions.

Variants vs version :

version = variability in time. The same software evolves in time. (not a software product line)

variant = variability in space. At the same time, you have several variants that exist.

Product derivation : you generate different product lines.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| domain engineering  development | domain analysis | domain implementation |
| application engineering | requirements analysis | product derivation |

Feature model

* features
* has a hierarchy : rooted tree
* notations for variability ; optional, alternative…

o plein : mandatory

o vide : optional (you can remove it)

A plein : or

A vide : alternative (like a xor)

* describes constraints.

domain implementation

* you implement assets : implementation in terms of source code, models, tests...
* You also have to do the mappings (to which feature is related the source code)
* It is reusable : you want to use it for different variants.

#ifdef en c

munge processor en Java

* included in swing library
* featureIDE tool

Colored IDE (CIDE) :

* colors each feature

compositional approach :

* you decompose your approach
* AHEAD framework

for each feature, you have an implementation

features

> feature 1

>> file.java

>> file.java

> feature 2

>> file.java

..