トップエスイー修了制作

AspectCによる組込みプロダクトライン開発

株式会社東芝　羽柴瑠弥子　rumiko.hashiba@toshiba.co.jp

開発における問題点

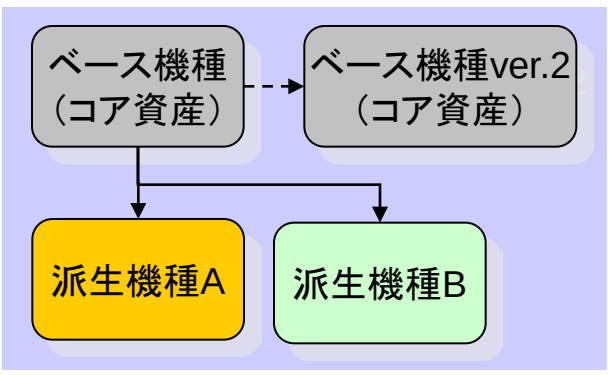
派生製品開発では、開発工数削減のため、コア資産を流用した製品開発を行っている。
しかし、コア資産を流用することで、ソースコードレベルにおいて2つの問題が発生する。
(1)コア資産と機種固有処理がもつれ合う問題
(2)機種固有処理が複数モジュールにちらばる問題

手法・ツールの提案による解決

派生機種固有の拡張を、C言語向けアスペクト指向プログラミング（AOP）コンパイラであるAspeCt-oriented C compiler（ACC)によって記述することで、コア資産と機種固有処理のもつれ合い（1）と、機種固有処理のちらばり（2）の2つの問題を解決し、組込み機種開発でのACC実用性について検証した。

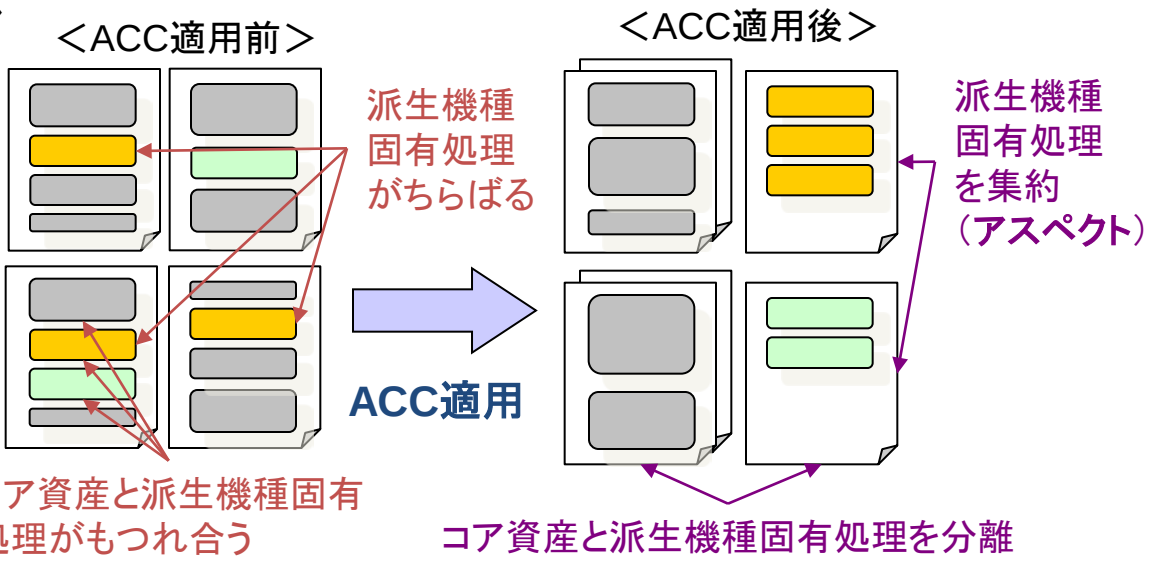
派生機種開発における課題とACC適用の効果

コア資産となるベース機種から、派生機種の開発を行う際、全ての機種処理がコンパイルスイッチにより同一ソース内で管理される。



⇒ベース機種開発の妨げになる。
⇒派生機種固有の処理を追いにくい。

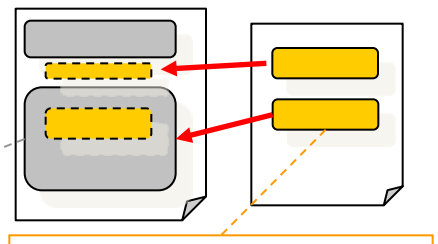
提案手法適用前後でのソースコードの変化



ACCの適用例

派生機種Aの処理を**アスペクト**として記述することで、コア資産を直接編集せずに「core_function()」に処理を差し込むことができる。

差し込みたい



```
・・・
int core_function(){
/* base source */
int ret = 0;
ret = funcA();
return ret;
}
```

```
before () : call($
core_function(...)){
funcC(); }
around():execution(int
core_function()){
int nret = funcB();
proceed();
}
```

組込みでの実用性

効果
- コア資産の保守性が向上
- 機種固有処理の可読性が向上

⇒ 派生機種を開発しながら、ベース機種が発展する製品開発に向いている

ACCの課題
- 生成されるソースコードのステップ数が増大
- コンパイル後のモジュールサイズが増大
- 入出力ファイルの再編集が必要
  - 入力ファイルから#ifdefの定義を削除
  - 出力のinline定義を削除

⇒ 組込み開発には不向きであるが、今後のACC発展によって改善する可能性がある

Aspect指向プログラミング（AOP）導入の課題
- AOP開発の教育コストが必要
- AOP記述確認により開発工数が増大