経過報告

田中 隆己

2012 年 4 月 17 日

1 HIME

HIME の PMT が落下する前にフレームにマウントしてしまいたい。フレームにマウントする前の作業としては、

- 遮光 (エッジ部分の遮光はこちらでやることにした)
- 遮光チェック

がある。ただ、これらの作業をやっている時間はないので、先にフレームにマウントしてしまおうかと思っている。

2 ANAROOT のバグ取り

ANAROOT の構造に関わるところに関してだいぶ納得がいく感じになったので、やっと解析に入れるようになった。まずは現在のコードの吐いている物理量が正しいかどうかチェックしている。以下はその過程で見つかったバグ。

- HOD で timing の calib. para. と offset para. がコード的に入れ替わっていた。(tu_cal, tu_off, td_cal, td_off) → (tu_cal, td_cal, tu_off, td_off)
- HOD の SetQCal とかの引数が int になっていた。
- NEBULA のクロストークイベントを拾うアルゴリズムで変数を間違えてぐちゃぐちゃな結果になっていた。

3 ANAROOT に関連した戯れ言

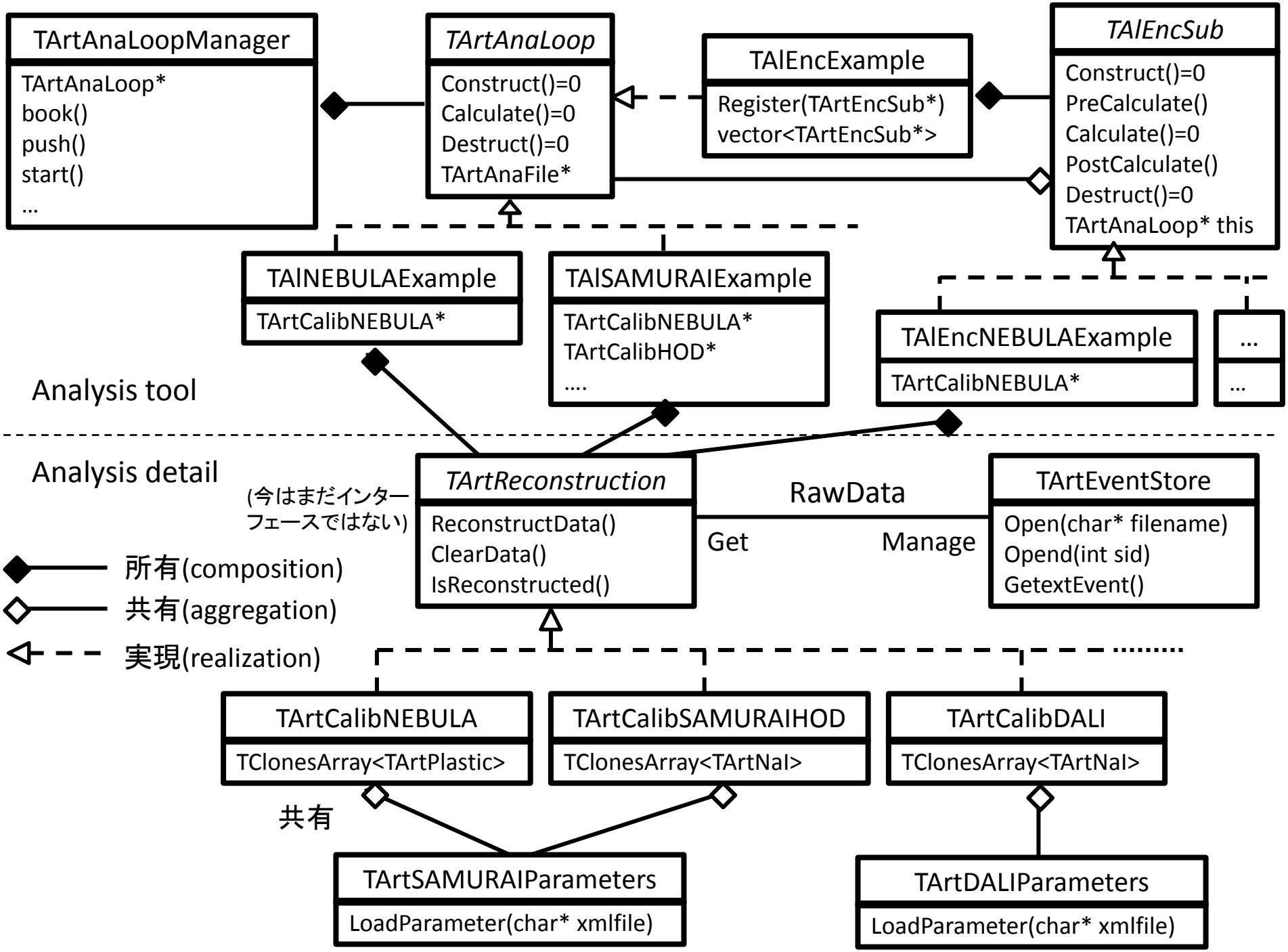
ANAROOT はオブジェクト指向を取り入れた解析ソフトである。

手続き型のプログラミングのみ勉強した人の多くはオブジェクト指向という抽象的な思考法を何かものすごく便利な方法と勘違いしている節があるが、あくまでプログラムを簡単に開発できるようにするための設計方法の指針の一つである。オブジェクト指向の便利さが理解いただけない人は、オブジェクト指向を実現するための技法によってプログラミング自体が人間の直感に近づくこと (「検出器」や「解析ルーチン」そのものが「生成」できる)、単純にコーディングが楽になること (標準ライブラリが充実 (配列の管理、ソート、検索)、二

回書かなくて良い (変更が極小、DEBUG 量が減る)) 等の直接的な利点に目を向けるべきである (本当のプログラマはうんぬんと言われるように、原理的にはオブジェクト指向で書いたプログラムは output として手続き型のプログラミングで実現できる)。

ANAROOT は解析ツールであり、最終的な output のみを提供する物ではない。そういう意味で、独自の解析ルーチンを追加したりする方法がコードレベルで簡単であると便利であるのは言うまでもない。例えば ANAPAW では新しい解析ルーチンを作るには enc.f を新しく書くだけでなく、usersrc.f を変更したり再コンパイルが必要であった。ANAROOT ではオブジェクト指向を実現するための技法によって、独自の解析ルーチン (ANAPAW でいう enc 何たら) を本体のソースを書き換える・再コンパイルすることなく追加することができ、さらに解析ルーチン自体を動的に生成・登録できるようにした。

オブジェクトの概念は人間が直感的に理解しやすいようにするためのものであり、オブジェクトの関係性を理解するのに難しく考える必要はない (これを実現するための技法は勉強する必要がある)。参考に、ANAROOT におけるオブジェクトの関連性を (似非)UML で示す (巻末)。


TArtAnaLoopManager
TArtAnaLoop*
book()
push()
start()
…
TArtAnaLoop
Construct()=0
Calculate()=0
Destruct()=0
TArtAnaFile*
TAlEncExample
Register(TArtEncSub*)
vector<TArtEncSub*>
TAlEncSub
Construct()=0
PreCalculate()
Calculate()=0
PostCalculate()
Destruct()=0
TArtAnaLoop* this
TAlNEBULAExample
TArtCalibNEBULA*
TAlSAMURAIExample
TArtCalibNEBULA*
TArtCalibHOD*
….
TAlEncNEBULAExample
TArtCalibNEBULA*
…
…
Analysis tool
Analysis detail
(今はまだインターフェースではない)
TArtReconstruction
ReconstructData()
ClearData()
IsReconstructed()
RawData
Get
Manage
TArtEventStore
Open(char* filename)
Opend(int sid)
GetextEvent()
所有(composition)
共有(aggregation)
実現(realization)
TArtCalibNEBULA
TClonesArray<TArtPlastic>
TArtCalibSAMURAIHOD
TClonesArray<TArtNaI>
TArtCalibDALI
TClonesArray<TArtNaI>
共有
TArtSAMURAIParameters
LoadParameter(char* xmlfile)
TArtDALIParameters
LoadParameter(char* xmlfile)