Detector characterization の体制

端山和大

Detector characterizationサブシステム構成

KAGRAチーフ会議

- **DetChar全体の進捗状況の報告**
- **DetCharチームの要望の訴え**
- **KAGRAからの意見、要請を受け取る**

⇕

ミーティング

- 基本的には全員参加
- 各リーダーがその週の進捗状況を報告
- アナウンスと議論（チーフ会議、学会など）
- 随時、方向修正
- それぞれの要望を出す

⇕

DetChar Projects		
Glitch Monitor	Line Monitor	...
リーダー	リーダー	...
解析メンバ	解析メンバ	
実験メンバ	実験メンバ	
DETメンバ	DETメンバ	

ソフトウェアの開発

システムとツールに分けて開発

システム

DetCharシステム上で常時動作している。基本的に一度立ち上げたら、触らない。なるべく安定動作、リアルタイム性を重視。

ツール

DetCharシステム、または単独でユーザが手動で用いる。

始めにツールとして開発を進めて、それから一部の基本的で重要なものをシステムとして組み込む。

システム

- DetCharシステムで動作すること
- リアルタイムで処理できること
- シンプルかつ安定なこと
- 使い方、サンプルコード、アルゴリズム資料があること

ツール

- 各自、自分の好きな言語で開発
- Cで書く場合、LAL、KAGALYに準拠
- リアルタイム性は追求しないが、遅すぎないこと
- インタプリタ言語で使えるとよい
- 使い方、サンプルコード、アルゴリズムの資料があること
- インプット、アウトプットインターフェースは統一する

ミーティング

- Weekly Meetingで各チームリーダーがプロジェクトの進捗状況の報告
- 新しいテーマの提案
- 問題点の報告
- Action Itemの管理
- プロジェクトのリプランニング
  （プロジェクトの運用は全員素人と考えて、ミーティングで話し合って頻繁に軌道修正を行って行く。）

DetCharプロジェクト

Primary projectとSpecial projectに分け、P-が本筋で、S-は短期的または本筋に入れるために検討するプロジェクト。P-を優先して進め、S-は適時チームを結成して進める。

各プロジェクトにリーダーを配置し、中心になって進める。

Primary Projects

- Detcharシステムの構築(incl. web-based)
- Glitch Monitor
- Line Monitor
- Gaussianity Monitor
- Noise Budget
- Health Monitor
- Data base
- Quality flag

Special Projects

- Globally correlated noise
- Violin mode
- Multi-Channel Analysis
- Observation shift

チーム構成

各プロジェクトは

- リーダー
- 解析グループメンバ ≧ 1人
- 実験グループメンバ ≧ 1人
- DetCharメンバ ≧ 1人

で構成する

Glitch Monitor

- **Glitch detection**
- **Statistics (frequency,..)**
- **Characterization**
- **Coherency check between channels**
- **Event display**

Glitch Monitor開発に必要なもの

- **Glitch detection**
  - **基本的なアルゴリズムはExcess power method**
  - **Omega、Omicron、KleineWelle、 ExcessPower、GlitchMonなどがある。**
- **Statistics**
  - **Glitchの頻度、時間帯など**
- **Characterization**
  - **duration, central frequency, power**
- **Coherency check**
  - **ピアソンの積率相関関数など**
- **解析結果の表示、グラフ**

Line Monitor

- **Line detection**
- **Statistics (frequency,..)**
- **Characterization (duration, central frequency, power)**
- **Coherency check between channels**
- **Event display**

Line Monitor開発に必要なもの

- **Line detection**
  - **基本的なアルゴリズムはSFT、Monochromatic wave fitting、HHT(?)**
  - **Fscan、LineMon、MBLTなどがある。**
- **Statistics**
  - **Lineの頻度、時間帯など**
- **Characterization**
  - **duration, central frequency, power**
- **Coherency check**
  - **ピアソンの積率相関関数など**
- **解析結果の表示、グラフ**

Gaussianity Monitor

- Noise floor tracking
- Power spectrum
- Rayleigh distribution tracking
- Realtime noise modeling
- Monitor display

Gaussianity Monitor開発に必要なもの

- Noise floor tracking
  - Noise floorのスペクトルを追う
  - NoisefloorMonなどがある。
- Power spectrum
- Rayleigh distribution tracking
- Realtime noise modeling
  - まずはStudent-t分布によるフィッティング
- Monitor display

Noise Budget

- Tools for obtaining transfer functions
- Understanding noise budget tools in LIGO

DetChar system

- **Platform to run the monitors automatically**
  - **DMT、Web-based displayなど**
- **Summary page generated automatically**
  - **定期的にwebに公開する**

Health Monitor

- To monitor important parameters of channels

チェックリストの活用

- ソフトウェアの開発はチェックリストを常時確認しながら行っていく。

チェックリスト（ツール開発）

- ツールの目的はなにか？
- 誰が使うのか？
- インプット、アウトプットフォーマットは規定に沿っているか？
- マニュアル、サンプルコードは用意しているか？
- コード内のコメントは修士１年生がわかるか？
- その箇所は共通ライブラリが使えないか？

チェックリスト（システムツール開発）

- ツールの目的は何か？
- 誰が使うのか？
- マニュアル、サンプルコードは用意しているか？
- リアルタイム解析できるだけ早いか？
- もっとシンプルにならないか？

タイムライン

近々の目標

- 秋の防振系テストでpreliminaryなツールが動かせること。
- 防振系テストで実験家に使ってもらいながらフィードバックをもらって改善する。

ルール、方針、心構え

- 常にチェックリストで確認しながら開発
- 人が作ったプログラムは使いにくいことを常に念頭に置いて、使う人の立場に立って開発
- プロジェクト間で共通で用いるアルゴリズム（プログラム）は、分かっているものについてはなるべく共有する。その場合、C/C++で書くのが良いだろう。

ツールのインターフェースについて

- インプットパラメータとアウトプットパラメータはどのツールも同じものにする。
- 途中の議論でインターフェースを変更する可能性もあることをあらかじめ承知しておく。