ゲームシステムのアーキテクチャと開発環境

– 開発効率と処理効率のために –

2014 年 4 月 16 日 第二稿

板垣 衛

■ 改訂履歴

稿	改訂日	改訂者	改訂内容
初稿	2014 年 3 月 13 日	板垣 衛	（初稿）
第二稿	2014 年 4 月 16 日	板垣　衛	「カメラ処理の効率化手法」の副題の誤字修正。 参考書籍を追記。

■ 目次

■ 概略

▼ 本書について

本書は、一連のドキュメントのインデックスです。

▼ ドキュメントの意図

各ドキュメントは、学習成果のまとめ、アイディアのまとめ、提案のまとめを意図したものです。それは、今後の技術的交流の円滑化のために、自身の技術・考えを他者に伝えることを意識したものです。

▼ ドキュメントの意識

各ドキュメントは、「ゲームプロミングへの応用」を強く意識しています。ゲームに特化しない一般的な技術も扱っていますが、必ず「具体的なゲームへの応用」を示すようにしています。

「参考書の写し」のような内容のものはなく、自身の考えを反映さえて、ゲーム開発に最適化するようにしています。

▼ ドキュメントの題目

各ドキュメントは、私自身の経験・得意分野（データ処理系を得意としています）・関心に基づいて題目を選出したものです。とはいえ、過去の実務上の取り組みを文書化するようなものではなく、また、その延長にあるようなものでもありません。

かつての経験・枠組みにとらわれることなく、反省を生かし、学習を重ねた上で、一層の最適化を目指したシステム設計と開発方法を考案し、まとめています。学習成果に基づく習作でもあります。

▼ ドキュメントの種類

ドキュメントの種類は下記のとおりです。

・ ゲームシステム設計
・ 開発環境／プロジェクト管理方法論

・ プログラミング技術

ドキュメントは、実用性を考慮し、題目ごとに小分けにしています。
本書では、各ドキュメントの要約を示します。

▼ ドキュメントの内容

各ドキュメントの多くは、基本的に「草案レベル」の内容です。

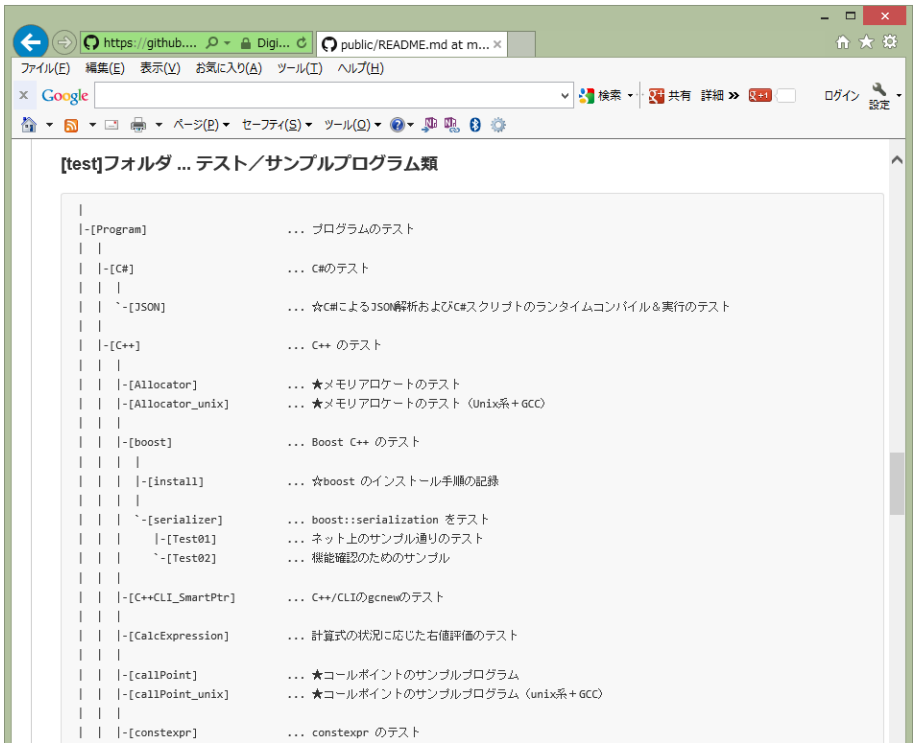
実証に基づいてまとめたものもありますが、確実性を得ていない内容も多く、そのようなドキュメントには「Draft」の透かしを入れています。

▼ サンプルプログラム

各ドキュメントにはサンプルプログラムを多数掲載しています。
幾つかのサンプルプログラムは、私個人の GitHub リポジトリから参照できます。

https://github.com/gakimaru/public

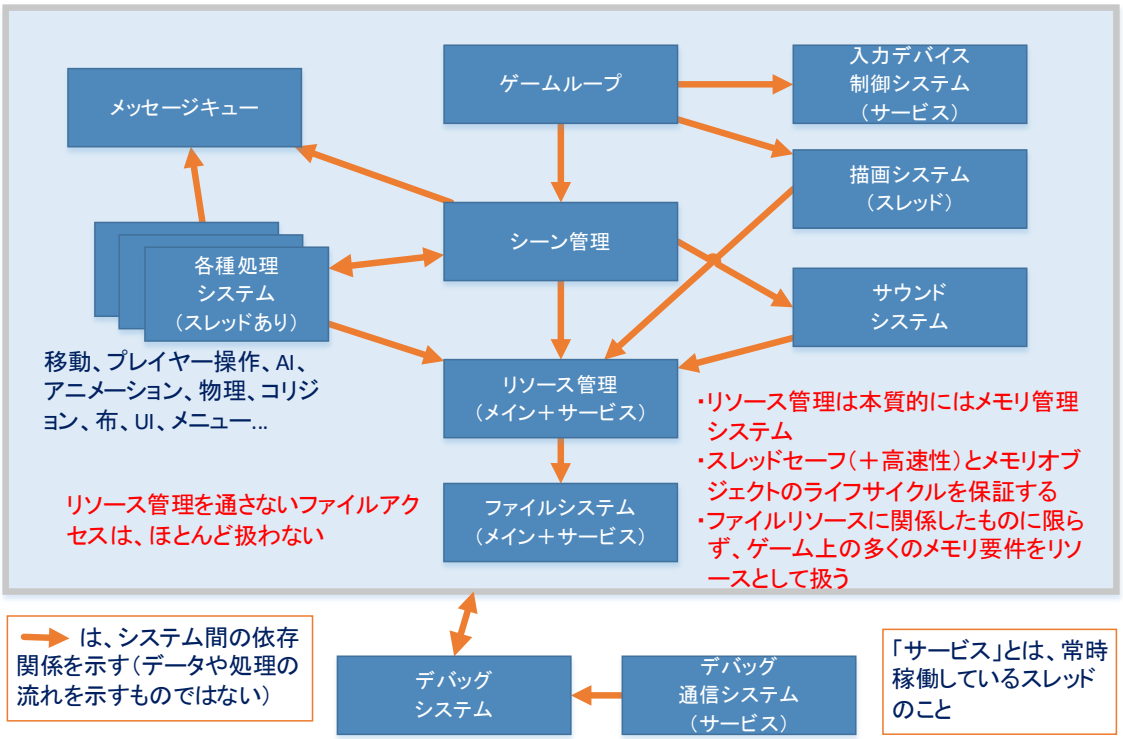
整然とドキュメントに対応した形でまとめているわけではおりませんが、リポジトリトップの README.md にて、どのようなサンプルプログラムを置いているか説明しています。



■ 基本ゲームシステム設計

幾つかのドキュメントでは、基本ゲームシステムの設計を示します。

▼ 基本ゲームシステムの全体像



▼ 基本ゲームシステム設計に関するドキュメント

基本ゲームシステム設計に関するドキュメントを示します。
各標題はそのままドキュメントのフォルダ名を示します。

● ゲームループ管理

- マルチスレッドによるゲームループ管理
  マルチスレッドに最適化したゲームシステムを統括

● ファイルシステム

- 開発を効率化するためのファイルシステム
  アーカイブファイルの効果的な活用

● リソース管理

- 開発の効率化と安全性のためのリソース管理
  マルチスレッドシステムを支える最重要要素

● シーン管理

- ゲーム全体を円滑に制御するためのシーン管理
  ゲームシステムの根幹
- オープンワールドのためのレベル管理
  シーン管理のサブシステムとしてのレベルコントローラ
- 効果的なイベントストリーミングシステム
  シーン管理のサブシステムとしてのイベントコントローラ

● サウンドシステム

- リソース管理を最適化するためのサウンドシステム
  サウンドマネージャの役割の明確化とシステムの連携

● デバイス管理

- 反応性と安全性を考慮した入力デバイス管理
  シンプルな処理で柔軟な制御

● デバッグシステム

- デバッグ制御システム
  コマンドベースの効率的なデバッグインターフェース作成
- ユニットテストと継続的ビルド
  コード修正の想定外の影響を早期に捕捉する

・ 効果的なデバッグログとアサーション
効果的なデバッグトレース手法

● ゲームデータ管理

・ ゲームデータ仕様
バージョン整合機能付きバイナリゲームデータ

・ ゲームデータ管理 DB システム
データベースによる安全で効果的なゲームデータ管理

・ ローカライズのためのテキスト管理構造
ゲームデータ管理 DB システムの活用

● AI

・ プランナーのための AI システム考察
「知識ベース」の活用

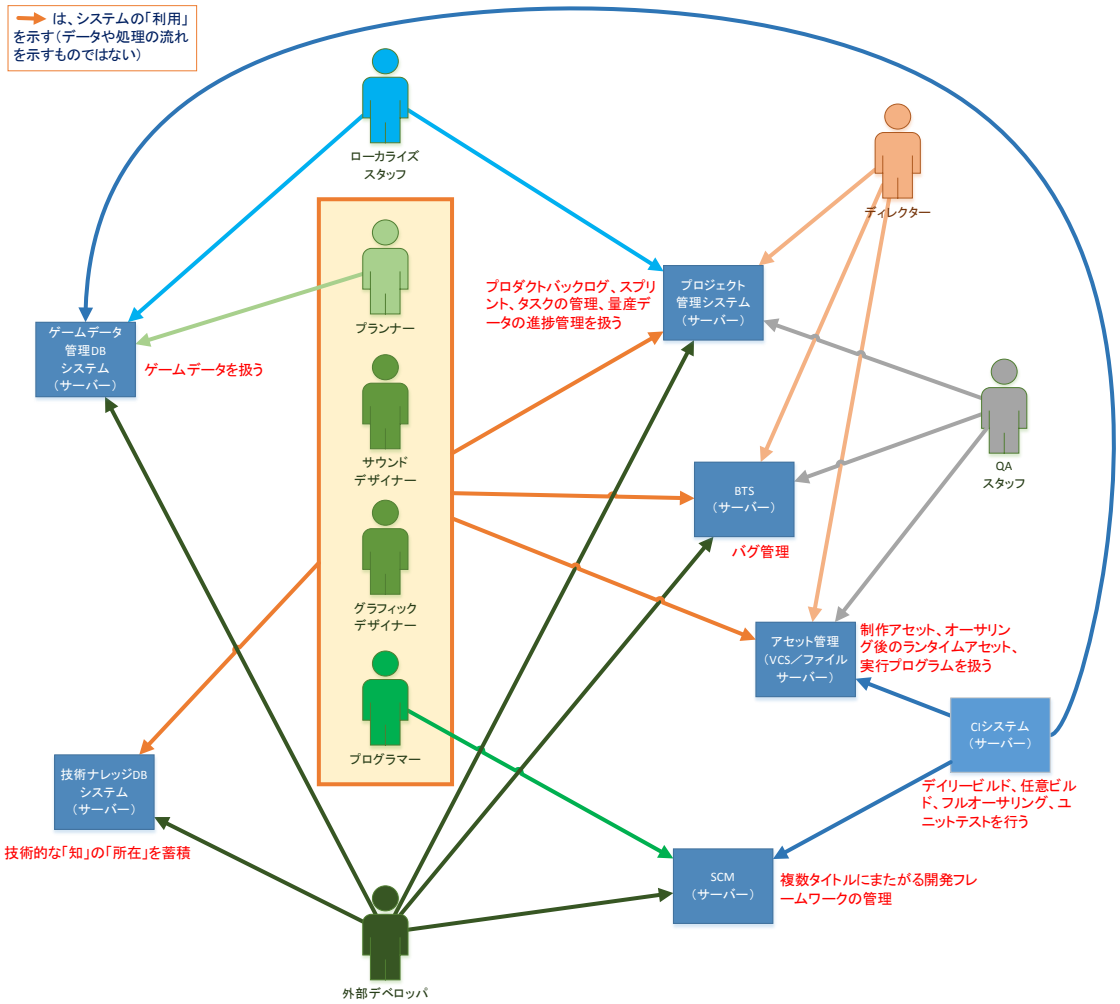
● その他

「プログラミング技術」に分類しますが、「メモリ管理」、「シリアライズ」、「サービス（スレッド）」もゲームシステムを担います。

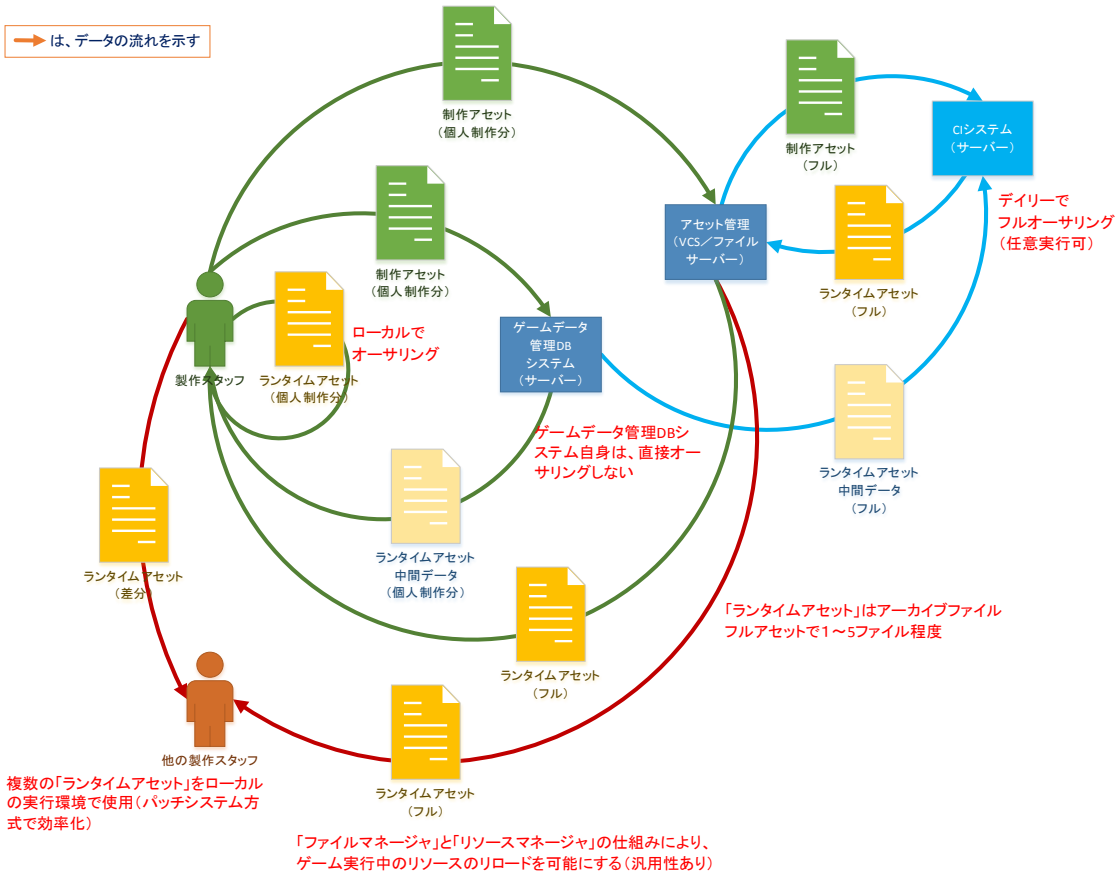
■ 開発環境／プロジェクト管理方法論

幾つかのドキュメントでは、開発環境とプロジェクト管理の方法論を示します。

▼ 開発環境の全体像（サーバーシステム）



▼ 開発環境の全体像（アセット管理）



▼ 開発環境／プロジェクト管理方法論に関するドキュメント

開発環境／プロジェクト管理方法論に関するドキュメントを示します。
各標題はそのままドキュメントのフォルダ名を示します。

● プロジェクト管理

- ゲーム開発のためのプロジェクト管理
  開発フェーズに合わせた柔軟で効率的な管理手法
- プロジェクト管理 Web システム
  開発フェーズに応じた管理システム

● 開発環境

・ 複数タイトルにまたがる効率的なフレームワーク管理
分散 SCM の活用

・ 技術ナレッジ DB システム
「知」の「所在」を明確にして無駄のない技術支援

● アセット管理

・ 効果的なランタイムアセット管理
差分アーカイブと自動リロードで制作効率を向上

● その他

「基本ゲームシステム設計」に分類しますが、「ゲームデータ DB システム」も開発環境を担います。

■ プログラミング技術

幾つかのドキュメントでは、プログラミング技術を説明します。

▼ プログラミング技術に関するドキュメント

プログラミング技術に関するドキュメントを示します。
各標題はそのままドキュメントのフォルダ名を示します。

● プログラミング Tips

・ チーム開発のためのコーディング手法
チーム開発に影響する安易なコーディングを改める

・ 本当にちょっとしたプログラミング Tips
共通認識にしたいプログラミングテクニック

・ オブジェクト指向と C++
オブジェクト指向言語としての C++とは

- プログラミング禁則事項
  従うべきプロジェクトのルールに配慮する
- 効果的なテンプレートテクニック
  ゲームプログラミングの最適化手法
- デザインパターンの活用
  デザインパターンをゲームプログラミングに役立てる
- プレイヤーに不満を感じさせないための乱数制御
  乱数の問題と性質を理解する

● マルチスレッド制御

- マルチスレットプログラミングの基礎
  最適なマルチスレッドプログラミングのために
- 効率化と安全性のためのロック制御
  より最適なマルチスレッドプログラミングのために
- 「サービス」によるマルチスレッドの効率化
  透過的な通信サービスとジョブスケジューラ

● メモリ管理

- ゲーム制御のためのメモリ管理方針
  ゲーム制御のためのメモリ管理の考え方
- 柔軟性を追求したメモリ管理システム
  ゲーム開発に効果的なメモリ管理
- 様々なメモリ管理手法と共通アロケータインターフェース
  メモリ操作の汎用化と共通ライブラリの有効活用

● シリアライズ

- セーブデータのためのシリアライズ処理
  互換性維持とデバッグ効率向上のために

● **スクリプト管理**

・ スクリプトの生産性向上のための名前付きデータ参照
画一化された手続きによるデータ参照手法

● **カメラシステム**

・ カメラ処理の効率化手法
デザインパターンの応用事例

■ 【課題】今後考えたい事

現状まとめきれていませんが、今後考えていきたい課題もあります。

▼ 品質向上のための仕組み

QA 向けに、品質向上のための情報収集のシステムを検討したいと考えます。
例えば、「A/B テスト」を応用したシステムに実用性を感じます。
「幾つかのパターンでプレイヤーが期待どおりの行動を行うのはどれか？」といったことを試行し、自動的に集計されるようなシステムがあると、品質向上に貢献できそうです。何にせよ、プレイの反応がプランナーに迅速にフィードバックされるような仕組みは考えたいところです。

▼ ゲーム内データベースの活用

ゲームの大規模化に伴い、ゲームに対する複雑なデータ要件をスマートに扱うことが求められます。
これまで、SQLite のような仕組みを用いるよりも、軽量化やデータの互換性維持の観点から、コンテナの仕組みをきちんと作ったほうがよいと判断してきましたが、今一度 SQLite に注目して複雑なデータ要件に対する生産性を検討してみたいと考えます。
また、C#の LINQ のような埋め込み SQL はとても生産性がよいものと感じます。（なお、LINQ は経験がありません。旧来の ODBC や ADO などはよく使用していました。）
ツールでもよいので、いっそプランナーが SQL を書くぐらいの想定で、複雑な AI 処理の開発に寄与する方法論を考えてみたいものです。

■ 参考資料

ドキュメントの作成にあたって、とくに参考にした書籍を示します。

・ 簡潔で再利用しやすいコードのための C++活用術

C++テンプレートテクニック

著者：επιστημη（えぴすてーめー）／高橋晶

発行所：ソフトバンククリエイティブ株式会社

特殊化、部分特殊化、高階関数、メタプログラミング、SFINAE、ポリシー、CRTP など、多数のテンプレート技術を学習。

・ **アジャイルなゲーム開発**

スクラムによる柔軟なゲーム開発

著者：クリントン・キース

訳者：江端一将

発行所：ソフトバンククリエイティブ株式会社

スクラムの実践方法の学習のほか、多くのゲーム開発事例を参考に。

・ **詳解 LINUX カーネル**

（第 3 版）

著者：Daniel P.Bovet

Marco Cesati

監訳者：高橋浩和

訳者：杉田由美子、清水正明、

高杉昌督、平松雅巳、

安井隆宏

発行所：株式会社オライリー・ジャパン

ゾーンアロケータ、ページフレーム、バディシステム、スラブアロケータ、メモリリージョンなどのメモリ管理、および、カーネル同期処理（spin-lock, read-write lock など）を参考に。

・ LINUX プログラミングインターフェース

著者：Michael Kerrisk

訳者：千住治郎

発行所：株式会社オライリー・ジャパン

SystemV の IPC ライブラリを参考に。

・ Git ポケットリファレンス

著者：岡本隆史、武田健太郎、相良幸範

発行所：株式会社技術評論社

Git の使い方を学習。

・ 開発ツール徹底攻略

（ムック本）

発行所：株式会社技術評論社

Git の管理構造を参考に。

・ 7 つの言語 7 つの世界

著者：Bruce A.Tate

監訳者：まつもとゆきひろ

訳者：田和勝

発行所：株式会社オーム社

Prolog を参考に。

・ 7 つのデータベース 7 つの世界

著者：Eric Redmond and Jim R. Wilson

訳者：角征典

発行所：株式会社オーム社

MongoDB を参考に。

・ **ガベージコレクションのアルゴリズムと実装**

著者：中村成洋／相川光

監修：竹内郁雄

発行所：株式会社秀和システム

Python のプールアロケータを参考に。

・ **Java で学ぶアルゴリズムとデータ構造**

著者：Robert Lefore

訳者：岩谷宏

発行所：ソフトバンク株式会社（1999 年当時）

赤黒木（red-black tree）を参考に。

・ **ゲームエンジンアーキテクチャ**

著者：ジェイソン・グレゴリー

監修：今給黎隆、桐山忍、鴨島潤、湊和久

訳者：大貫宏美、田中幸

発行所：ソフトバンククリエイティブ株式会社

フレームアロケータを参考に。

・ **ヘネシー＆パターソン コンピュータアーキテクチャ**

定量的アプローチ（第 5 版）

著者：ジョン・L・ヘネシー、デイビッド・A・パターソン

監訳：中條拓伯、天野英晴、鈴木貢

翻訳：吉瀬謙二、佐藤寿倫

発行所：株式会社翔泳社

CPU キャッシュ、命令パイプラインを参考に。

掲載は割愛しますが、以上の他にも多数の書籍、および、Web サイト（とくに Wikipedia）を参考にしています。

■■以上■■

ゲームシステムのアーキテクチャと開発環境

以　上