
グリッド協議会
2007年度 第4回 Grid Hotline

グリッドプログラミングモデルと
そのインタフェース

– OGF初のfinal recommendation成立の裏側 –

産業技術総合研究所 グリッド研究センター
田中良夫、中田秀基



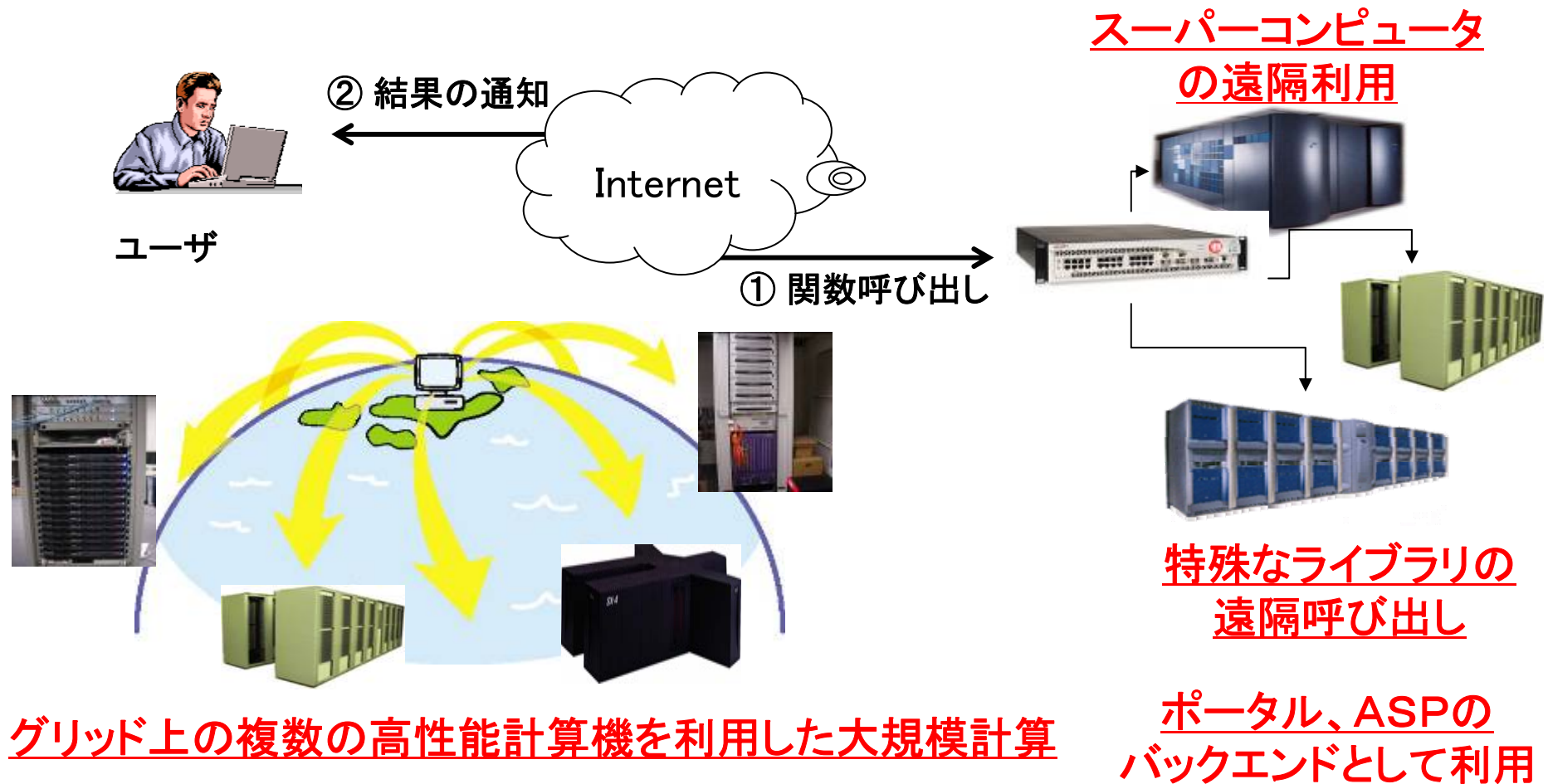
話の内容

- OGF初のRecommendationとなった”A GridRPC Model and API for End-User Applications” (GFD-R 52)について、その内容と最終標準への道のりを紹介。
 - ▶ OGF20@Seattleで発表
 - ▶ 9月27日に承認されていた
 - ▶ DRMAAとGridRPC APIの2本
- GridRPCとそのAPI
 - ▶ GridRPC: グリッドにおける遠隔手続き呼び出しに基づくプログラミングモデル
 - ▶ GridRPC API: GridRPCに基づくプログラミングを行うためのC言語のAPI
- 標準化への道のり
 - ▶ OGFにおけるドキュメントの種類とOGF文書へのプロセス
 - ▶ WGの立ち上げ～ドキュメント作成、審査、最終標準まで

GridRPC API標準ドキュメントの中身

GridRPC Model and API for End-User Applications
クライアントプログラミングのための標準API

GridRPC



GridRPC の構成要素

● クライアントコンポーネント

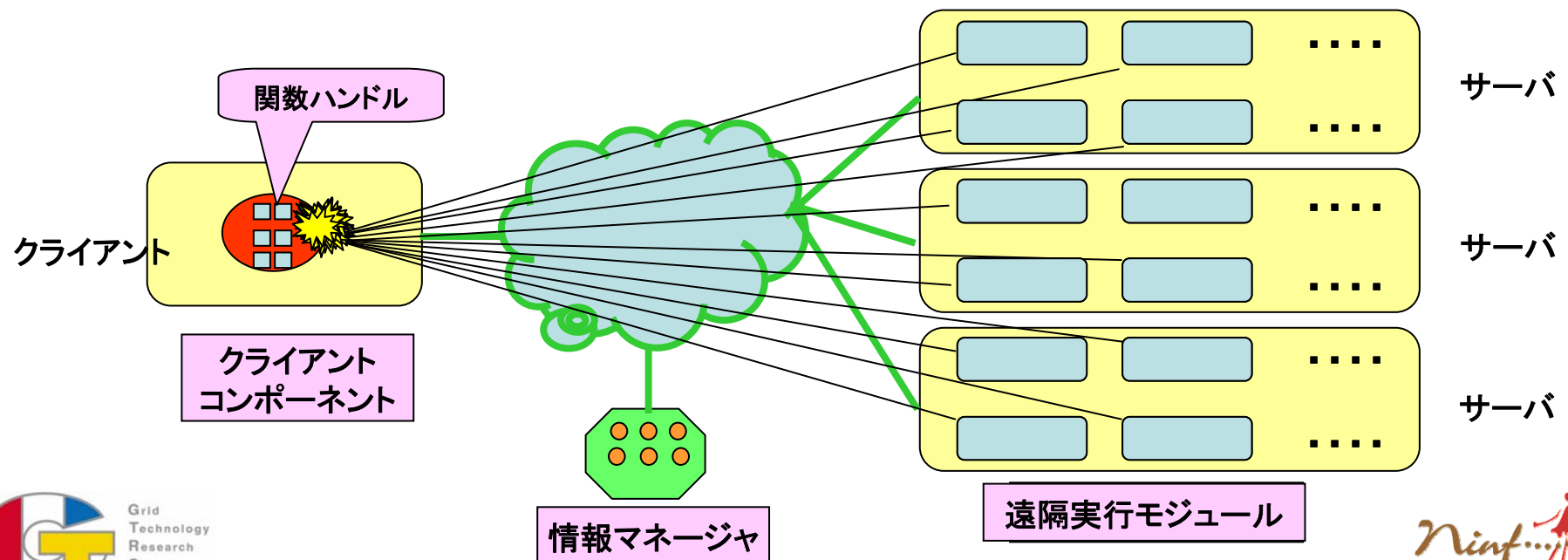
- ▶ GridRPCを用いてリモートライブラリを呼び出す側。
- ▶ 関数ハンドルを介して遠隔実行モジュールを操作・管理する。

● 遠隔実行モジュール (Remote Executables)

- ▶ GridRPCによってクライアントから呼び出される側。
- ▶ そのプロセスは動的に起動される。

● 情報マネージャ

- ▶ 遠隔実行モジュールに関する情報を提供・管理する。



GridRPC v.s. MPI

	GridRPC	MPI
並列性	タスク並列	データ並列
プログラミングモデル	クライアント/サーバ	SPMD
プログラミングAPI	GridRPC API	MPI
コアロケーション	必須でない	必須
障害復旧機能の実装	優れている	貧弱
プライベートIPノード	利用可能	利用不可
リソース割り当て	動的指定が容易	基本的に静的
その他	既存の逐次アプリケーションのグリッド化が容易	広く知られている 既存のMPIアプリがそのまま動く

GridRPCの現状

● Ninf、NetSolveプロジェクトは1994年頃スタート

- ▶ 当初はクライアント・サーバ型(1対1)の計算モデルを想定
- ▶ クラスタの普及に伴い、マスター・ワーカ型(1対多)の計算モデルが主流に

● 既存のシステムおよびその利用

- ▶ GridSolve @ UTK
 - ◎ 細胞生理学向けモンテカルロシミュレーション(MCell)
- ▶ DIET @ Inria
 - ◎ Workflowのバックエンドとしての利用を想定したシステム
- ▶ OmniRPC @ U.Tsukuba
 - ◎ HMCS-G (GRAPE-6 + CP-PACS)
- ▶ Ninf-G
 - ◎ 最新版はNinf-G Version 4
 - ◎ 大規模実証実験多数
 - ◎ Grid Interoperation Now (GIN)のテストアプリにNinf-Gアプリを利用

The GridRPC Model and API for End-User Applications

- RPCを用いた標準APIを提供。ポータブルかつ単純なプログラミングインタフェース(C言語のAPI)を提供。
- GridSolveやNinf-G等既存のシステム間での移植性を高める。
- GGF GridRPC WGにおける標準化
 - ▶ エンドユーザ向けAPIとミドルウェア向けAPIを分離
 - ▶ まずはEnd-User APIを標準化
 - ⊗ RPCに必要な機能のコアの部分のみ定義
 - ▶ ミドルウェアAPIは現在議論中
 - ⊗ データハンドル、スタックにより引数を受け渡すAPIなど
 - ▶ いくつかの参照実装を提供
- Out of Scope
 - ▶ Middleware API
 - ▶ Service Discovery
 - ▶ Non-flat Service Names
 - ▶ General Workflow
 - ▶ Interoperability between Implementations

Authors & Contents

Authors:

H. Nakada (AIST), S. Matsuoka (TITECH), K. Seymour, J. Dongarra (UTK)
C. Lee (Aerospace Corp.), H. Casanova (UCSD)

Contents:

1. Introduction
2. The Basic GridRPC Model
3. Document Scope
4. GridRPC API
 1. GridRPC Data Types
 2. Initializing and Finalizing Functions
 3. Remote Function Handle Management Functions
 4. GridRPC Call Functions
 5. Asynchronous GridRPC Control Functions
 6. Synchronous GridRPC Wait Functions
 7. Error codes and Error Reporting Functions
5. Related Work
6. Security Considerations



標準化の道のり

山あり谷あり地雷あり

OGFドキュメント

● OGFドキュメントの種類およびその承認手順に関するドキュメント

- ▶ Charlie Catlett, “Global Grid Forum Documents and Recommendation: Process and Requirements (GFD-C.1)”, June 2001, Revised April 2002
- ▶ Charlie Catlett, et.al., “Open Grid Forum Document Process and Requirements”に置き換わる予定。

Ⓢ 現在Public Commentが終わった段階

● OGFドキュメントの種類とその本数(2007年12月17日現在)

- ▶ Grid Working Draft (115本)

Ⓢ OGF Editorに投稿され、review processに入ったドキュメントの総称

- ▶ Informational GFDs (GFD-I) (63本)
- ▶ Experimental GFDs (GFD-E) (12本)
- ▶ Community Practice GFDs (GFD-C) (11本)

- ▶ Recommendation Track GFDs

Ⓢ Proposed Recommendation (GFD-R-P) (27本)

⊕ 十分なレベル、認知度にあるが、将来的に撤回の可能性もある。

Ⓢ Draft Recommendation (GFD-R-D)

⊕ 最低2つの実装により検証されている。

⊕ ドキュメントフローには、GFD-R.Dの状態が記載されていない！

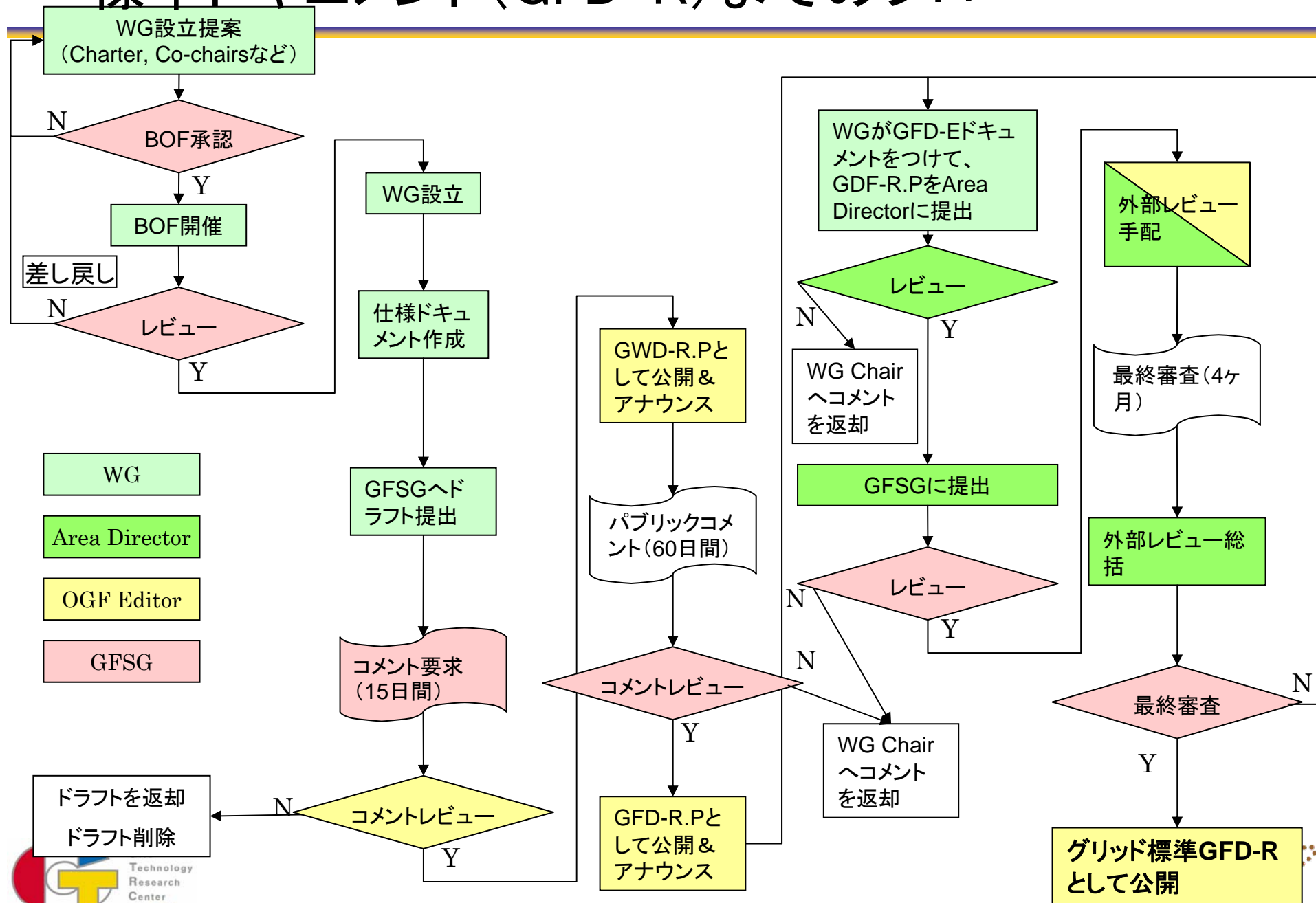
⊕ 実際にもGFD-R.Dの番号が振られたことはなかった！

Ⓢ Grid Recommendation (GFD-R) (2本)

⊕ 最終版



標準ドキュメント(GFD-R)までのフロー



WG

Area Director

OGF Editor

GFSG

GridRPC WG立ち上げに至る布石

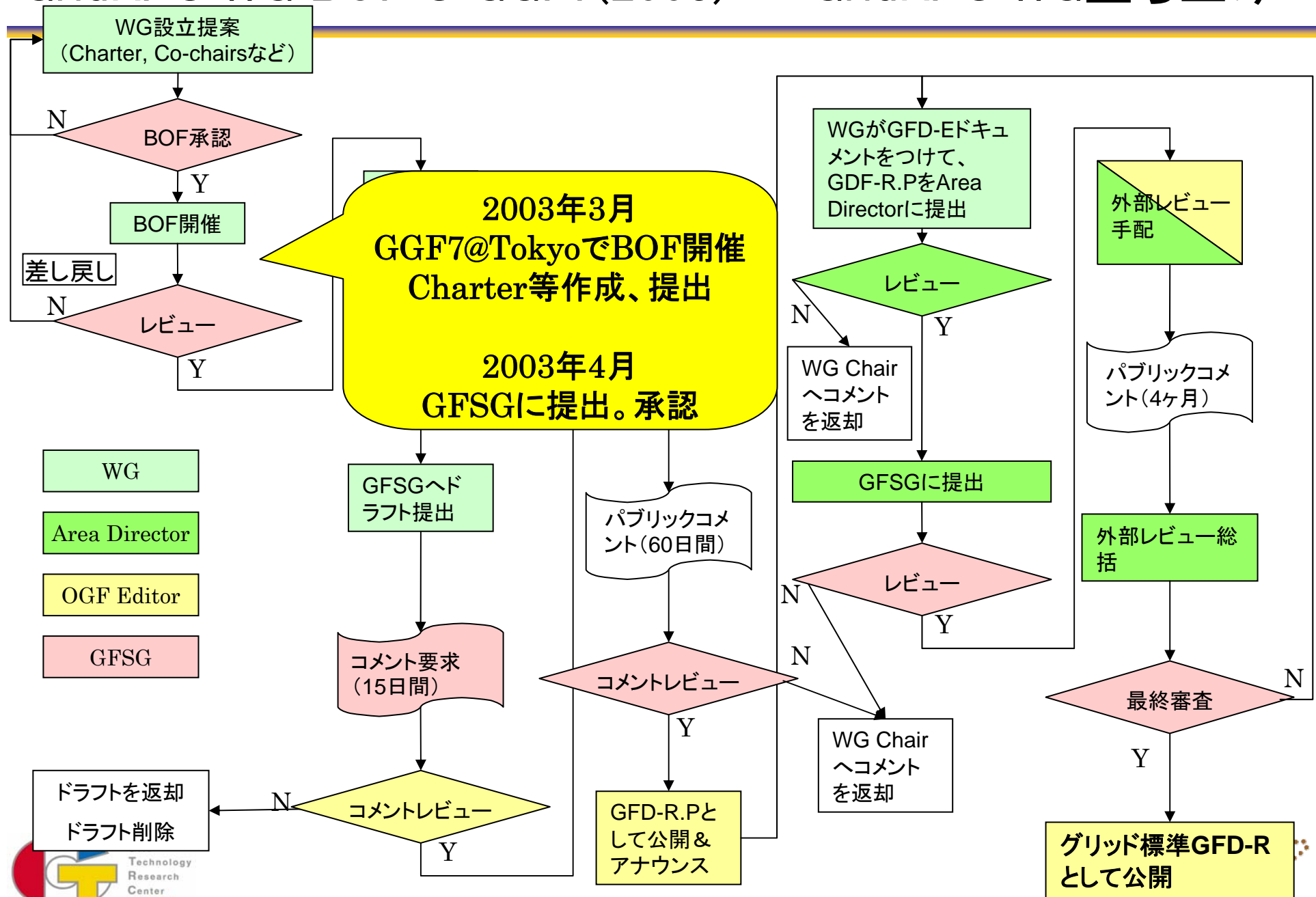
- 1994年頃からNinf@電総研とNetSolve@UTKの研究がスタート
 - ▶ 1996年から交流が進められていた。
- 1999年に始まったGrid Forumのときから、Advanced Programming Models WGにおいて、GridRPCを中心としたプログラミングモデルに関する議論を行っていた。
- 2001年のGGF発足時、GFSGに松岡先生@東工大と関口@産総研が入っていた。
 - ▶ 松岡先生はAPMEのArea Directorだった
- 2001年～2002年の時点で、GridRPCとそのAPIに関する発表をするまで進んでいた
 - ▶ 中田秀基、田中良夫、松岡聡、関口智嗣、“GridRPCシステムのAPIの提案”、ハイパフォーマンスコンピューティング研究会、2001.
 - ▶ Keith Seymour, Hidemoto Nakada, Satoshi Matsuoka, Jack Dongarra, Craig A. Lee, Henri Casanova:
Overview of GridRPC: A Remote Procedure Call API for Grid Computing.
IEEE Intl. Workshop on Grid Computing, pp. 274–278, 2002.
- 実装に基づいたAPI策定



- ▶ Interoperationはある程度見えていた



GridRPC WG BOF @ GGF7(2003) ~ GridRPC WG立ち上げ



GridRPC WG

- **Group Abbreviation:**

gridrpc-wg

- **Group Name:**

Grid Remote Procedure Call WG

- **Area:**

Applications

- **Co-Chairs**

- ▶ Craig Lee (The Aerospace Corporation) 2003年4月～
- ▶ Hidemoto Nakada (AIST) 2003年4月～
- ▶ Keity Seymour (UTK) 2003年4月～
- ▶ Eddy Caron (INRIA) 2007年5月～

- **Secretary**

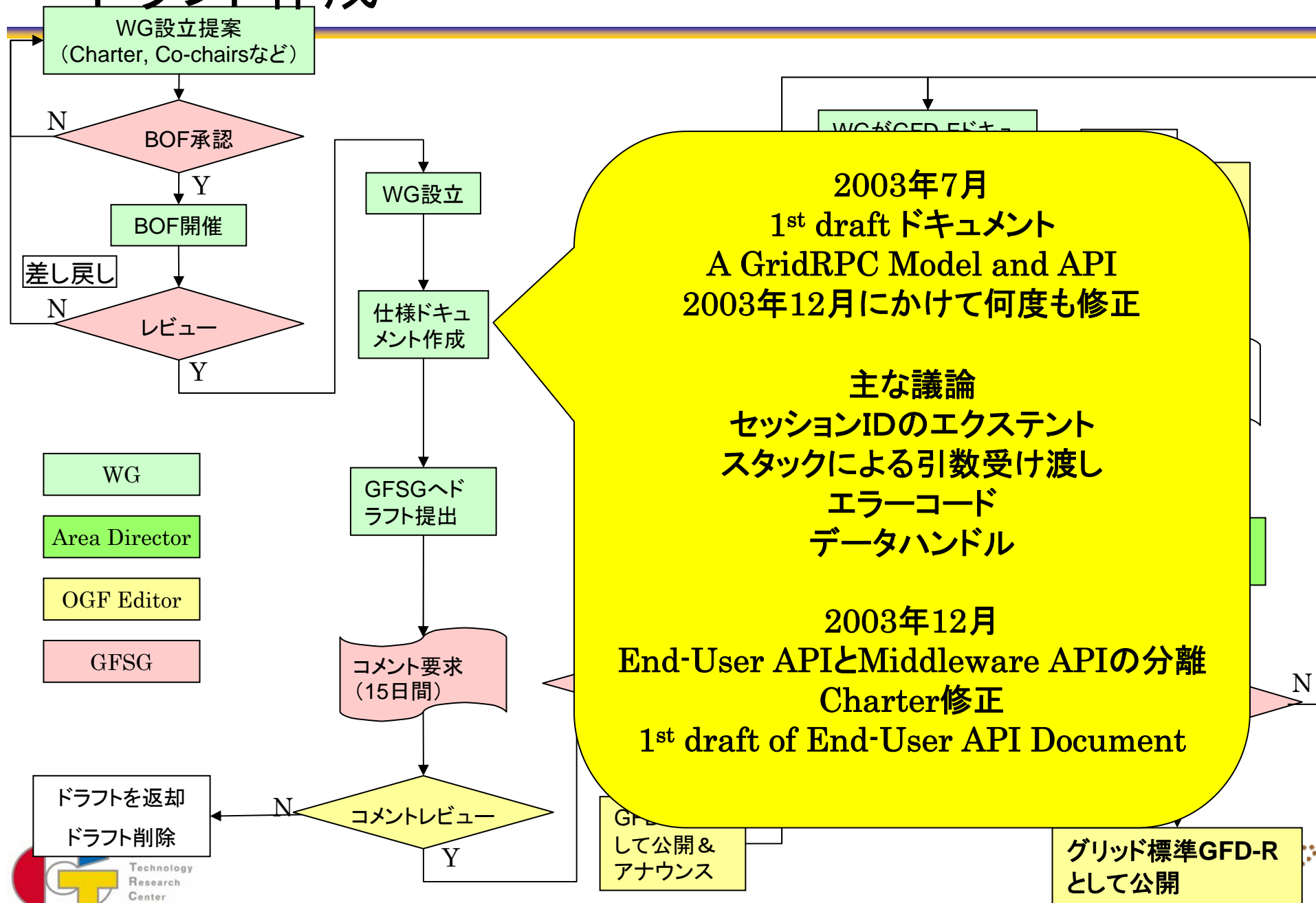
- ▶ Yoshio Tanaka (AIST) 2003年4月～

📧 谷村@AISTに引き継ぐつもり

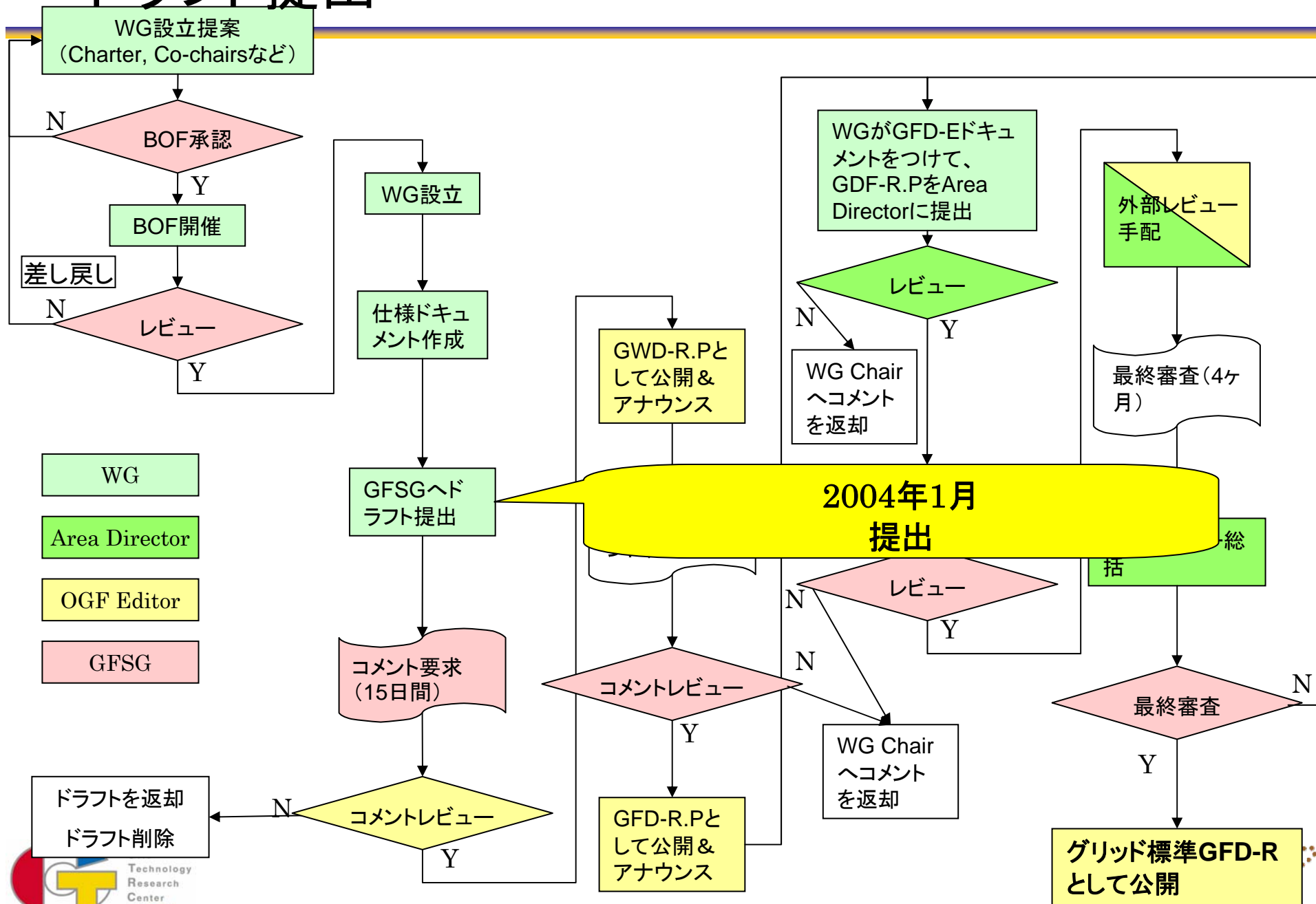
- **Goals and publications**

- ▶ A GridRPC Model and API for End-User Applications (REC) done
- ▶ A GridRPC Model and API for Middleware Developers (REC) ongoing
- ▶ Interoperability Testing for The GridRPC API Specification (EXP) done

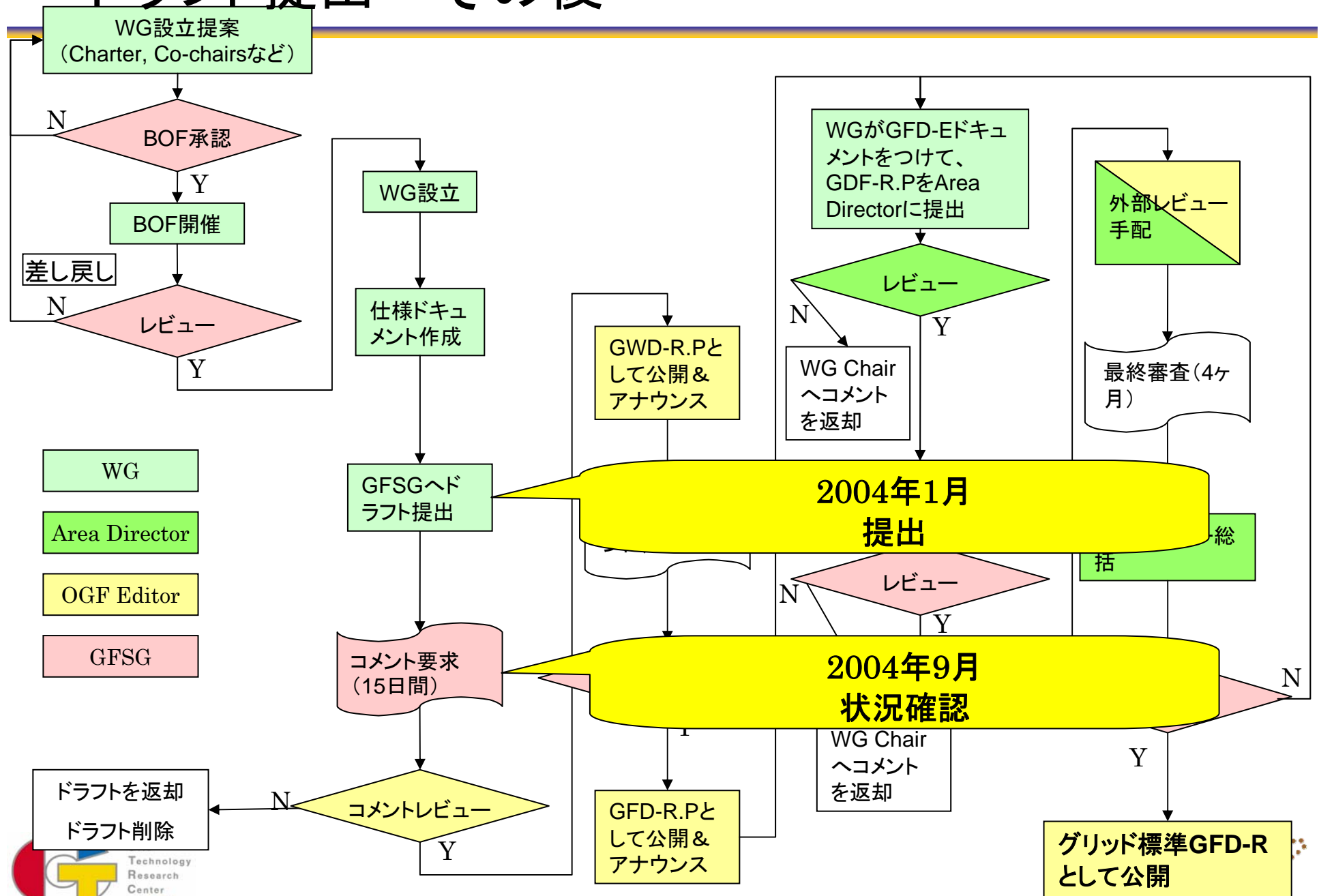
ドラフト作成



ドラフト提出



ドラフト提出～その後



WG

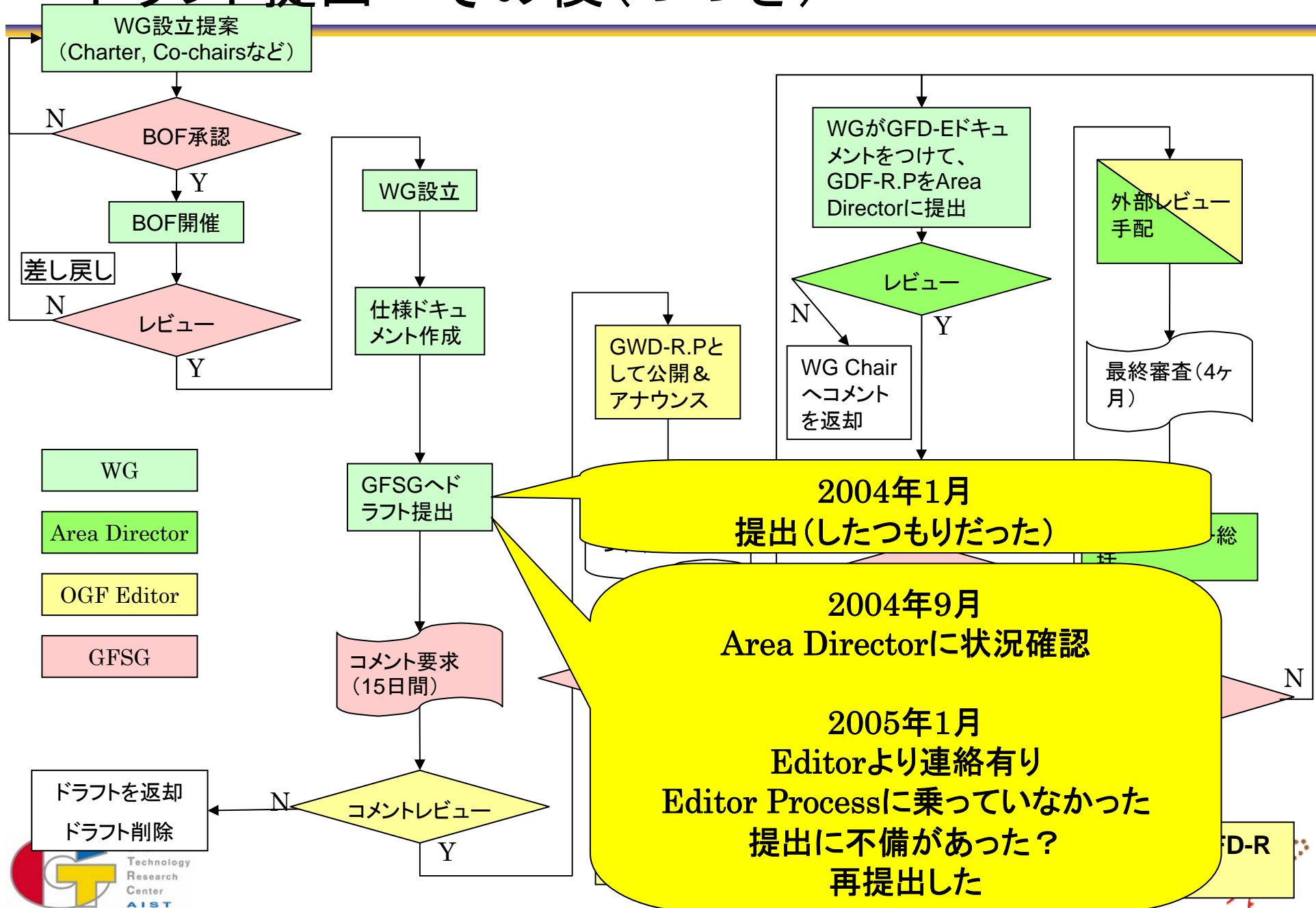
Area Director

OGF Editor

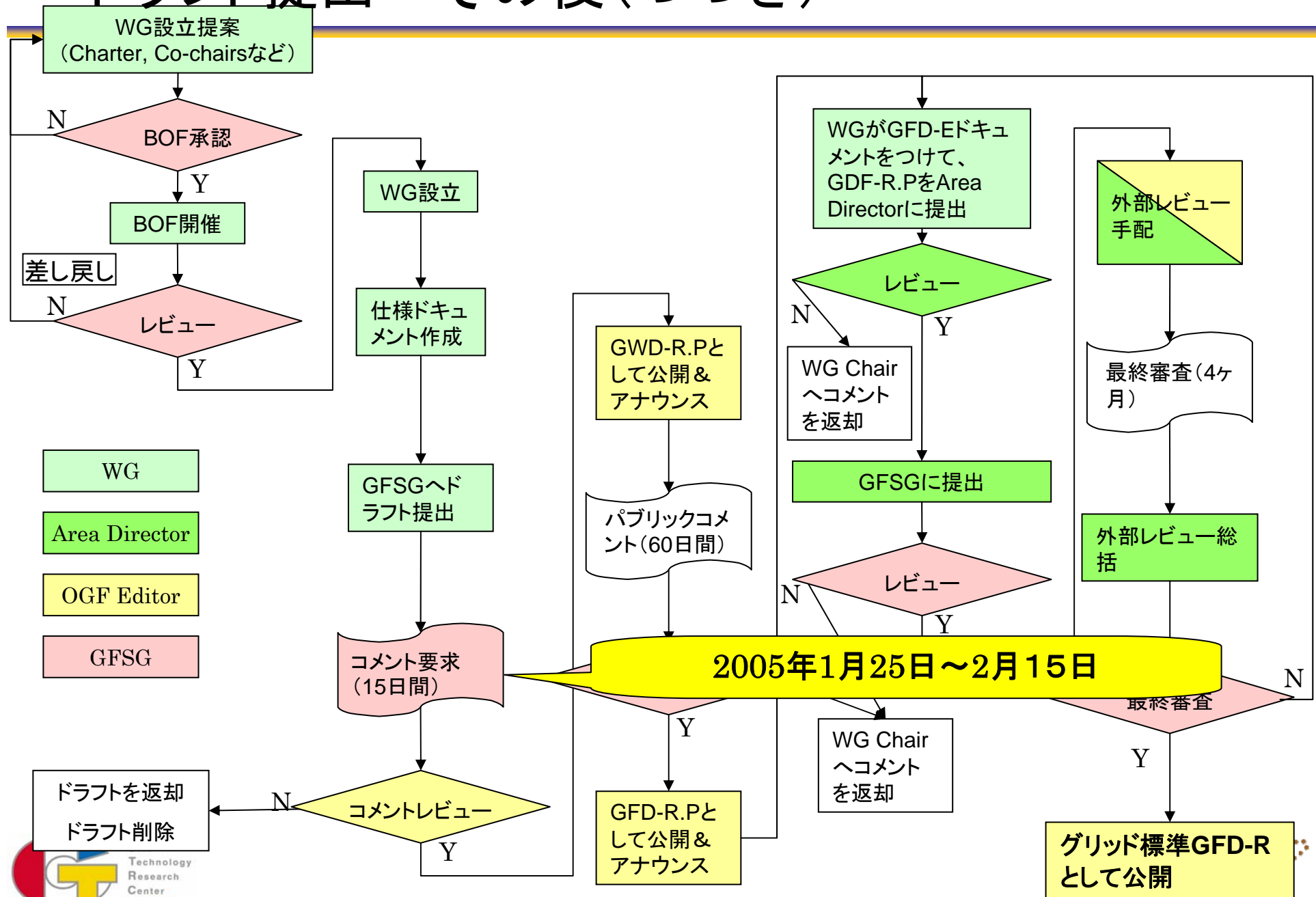
GFSG



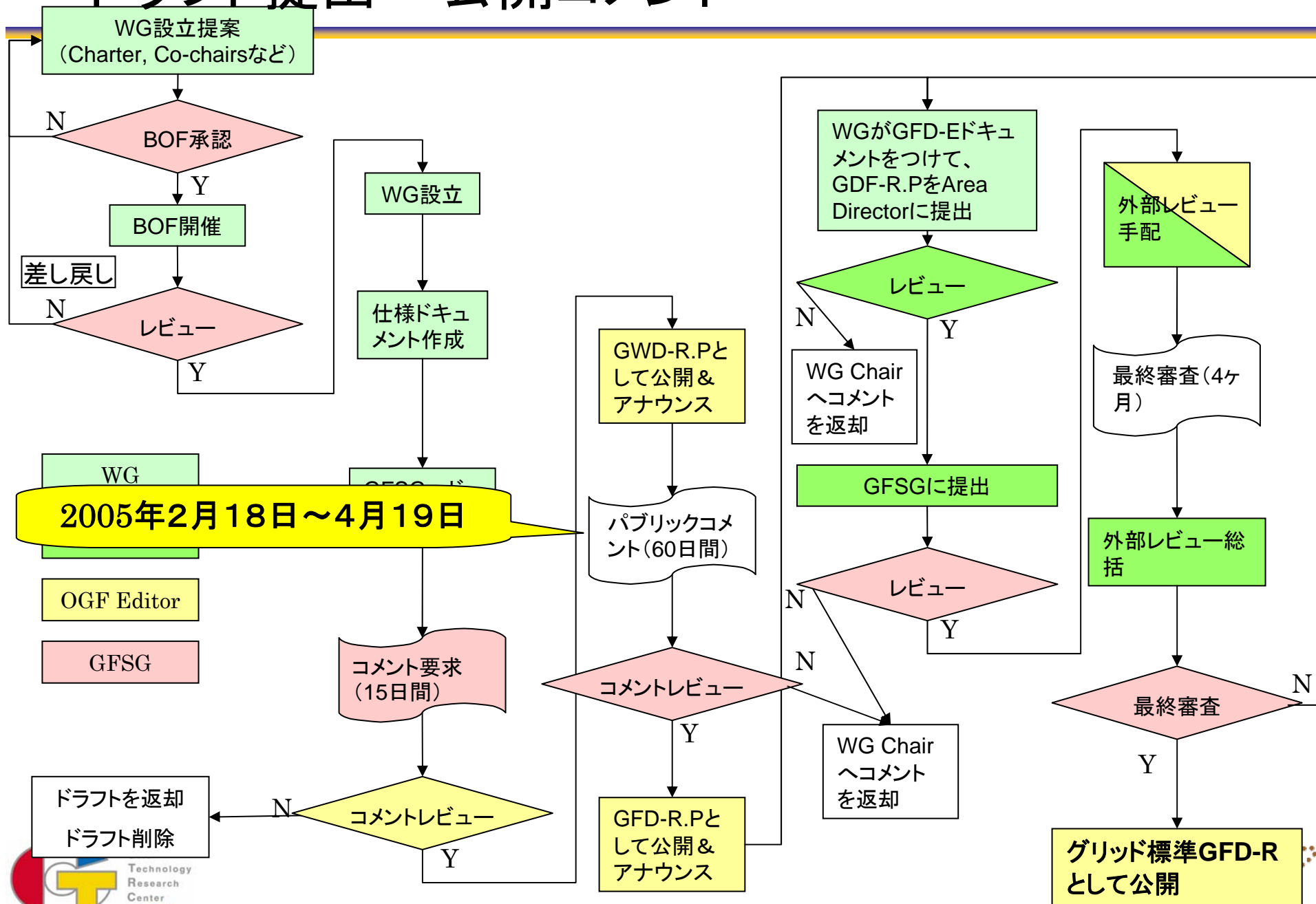
ドラフト提出～その後(つづき)



ドラフト提出～その後(つづき)



ドラフト提出～公開コメント

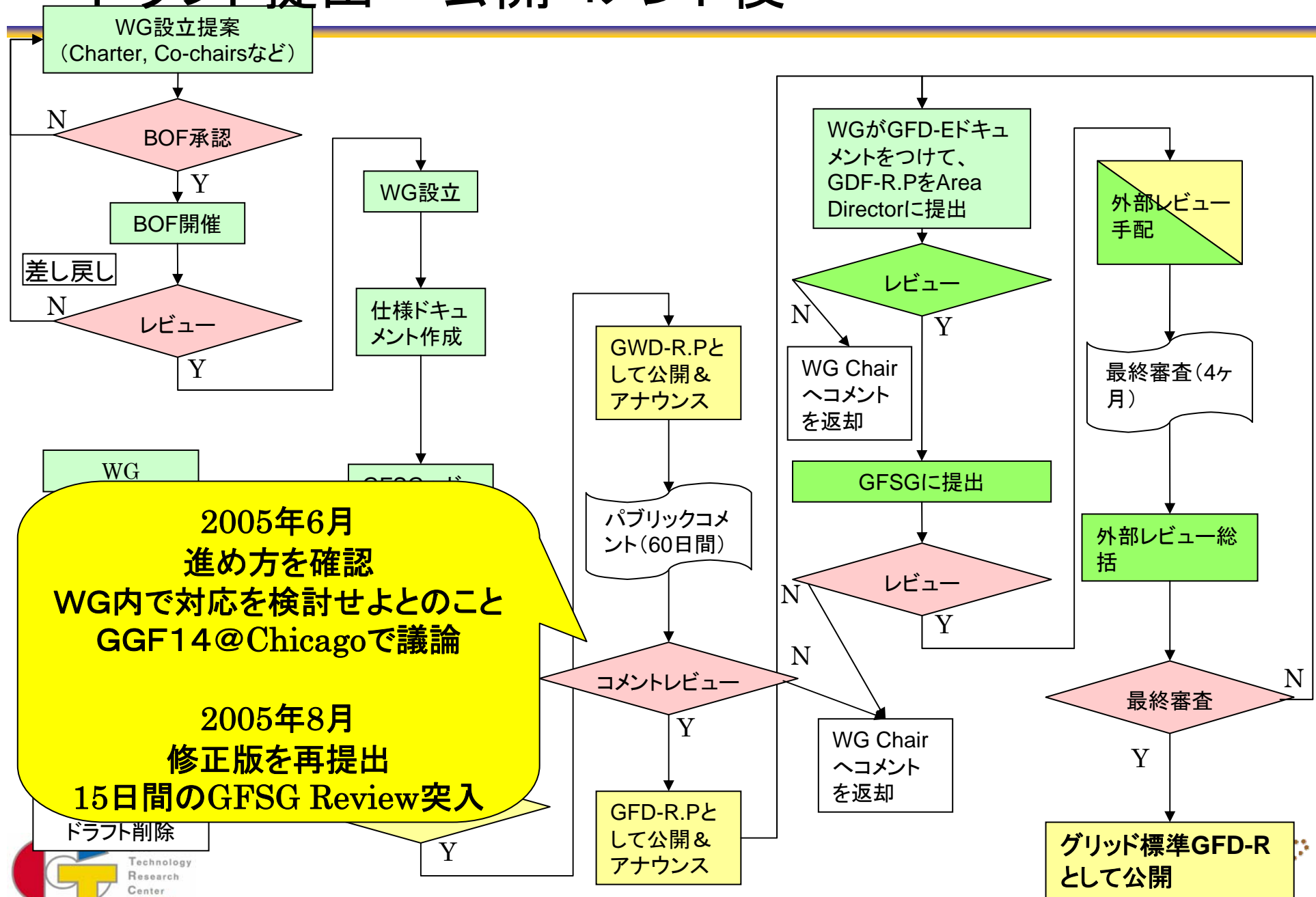


2005年2月18日～4月19日

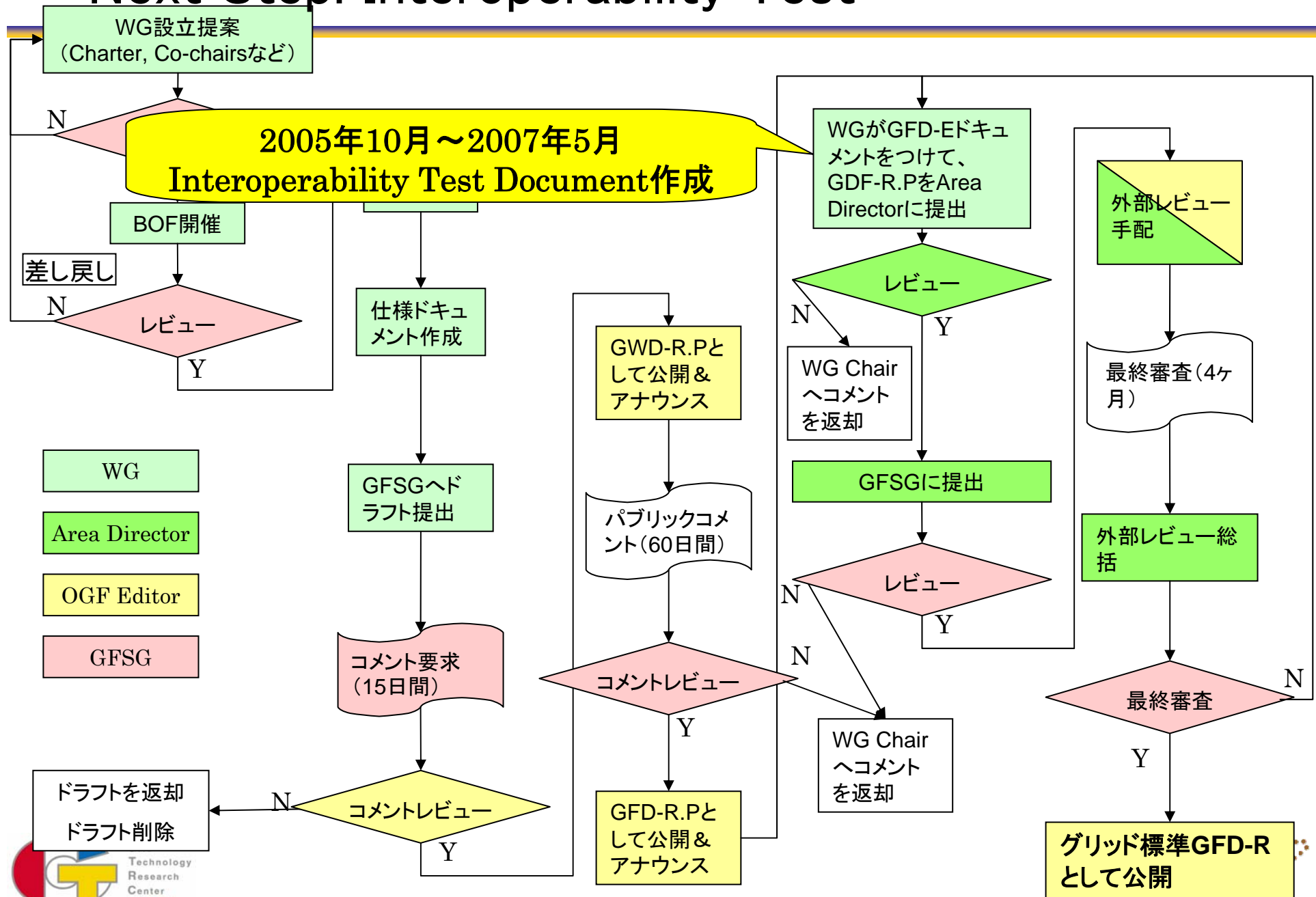
- WG
- OGF Editor
- GFSG



ドラフト提出～公開コメント後



Next Step: Interoperability Test



Interoperability Test Document

- 2005年10月～ やったこと
 - ▶ テストスイツ開発
 - Ninf-Gのテストスイツをベースにした
 - ▶ テスト、検証、実装修正
 - GridSolveチームの協力も得てNinf-GとGridSolveでテスト。
 - ▶ テストスイツのForgeへの登録
 - DIETのテスト (by INRIAチーム)

- 2006年7月
 - ▶ Area Directorにドキュメント提出
 - ▶ いくつかコメント

- 2006年10月
 - ▶ 修正後Editorに提出

- 2006年12月
 - ▶ (なぜか)ETSIよりコメント(GridRPC APIとByteIOに対して)

- 2007年4月
 - ▶ 修正版を再提出
 - ▶ Specification (GRD-R.P)のほうも少し修正

- 2007年5月
 - ▶ GFD-E.102として公開



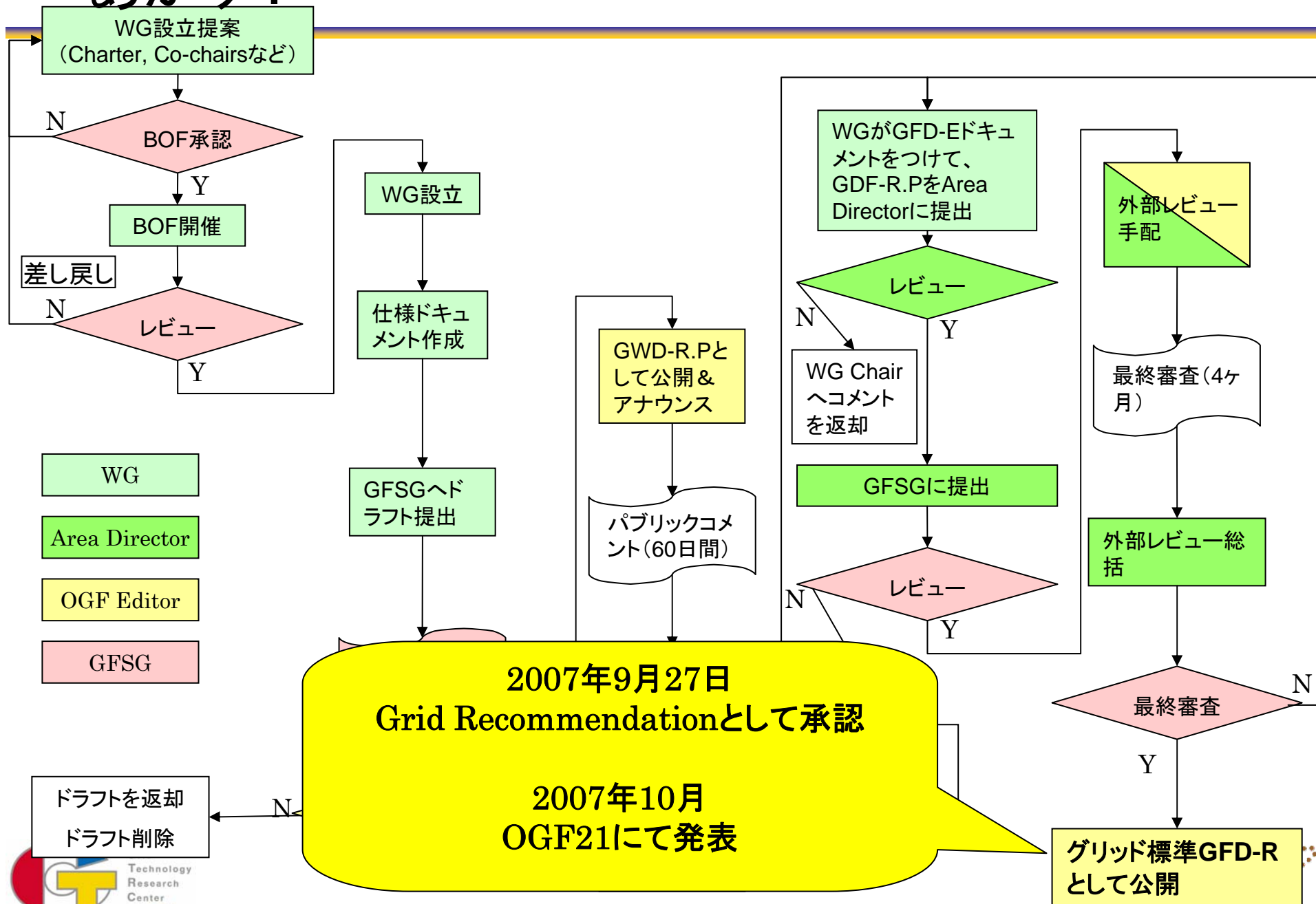
ETSIからのコメント

- GridRPC APIとByteIOの両方のInteroperability Testのreviewを行った。
- 双方の用語の使い方に統一性がない
 - ▶ GridRPC の”test cases”は「何をテストすべきか」
 - ⊗ ETSIは”test purposes”という用語を用いる
 - ▶ GridRPCの”test code”はETSIの”test case”。いずれも一種のExecutable。
 - ▶ ByteIOの”test code”はETSIの”test purpose”より、さらに詳細（表示されるメッセージなど）だが、それらはExecutableではない。ETSIでは、これらは”test description”と呼んでいる。（”test purpose”と”test case implementation”の間にある段階）
 - ▶ “test purposes”と”test case”に統一した
- テストの目的と方法を明確に記述すべき
 - ▶ ByteIOは混在している

ETSIからのコメント(つづき)

- Interoperability testingと書いてあるが、これはconformance testだと思う。なぜconformanceという用語を用いない？
 - ▶ GridRPCは複数の実装上でのクライアントプログラムの動作検証をしているということは、実際にはconformance testではないか？
 - ◎ OGFはconformance testはやらないとうたっているので、その用語は用いなかったといういきさつがあった。
 - ◎ 「Interoperability Testだが、実際にはConformance Testになっている」というようなことを書いた。
- システムを構成するエンティティ(例えばGridRPCのクライアントやサーバ)ごとにテストするべきでは？
 - ▶ 例えばNinf-G ClientがGridSolve Serverを呼ぶとか。
 - ▶ このテストではAPIレベルの互換性を検証していることを明確にした
- GridRPCとByteIOはInteroperability Testのレベルが違う
 - ▶ GridRPCはソフトウェアコンポーネント、ByteIOはプロトコルレベル
- GridRPCはなぜC言語だけなのか？
- コメントのまとめ
 - ▶ 目的と手段を明確にすること
 - ▶ 「必須」と「オプション」の区別を明確にすること

あがり！



WG

Area Director

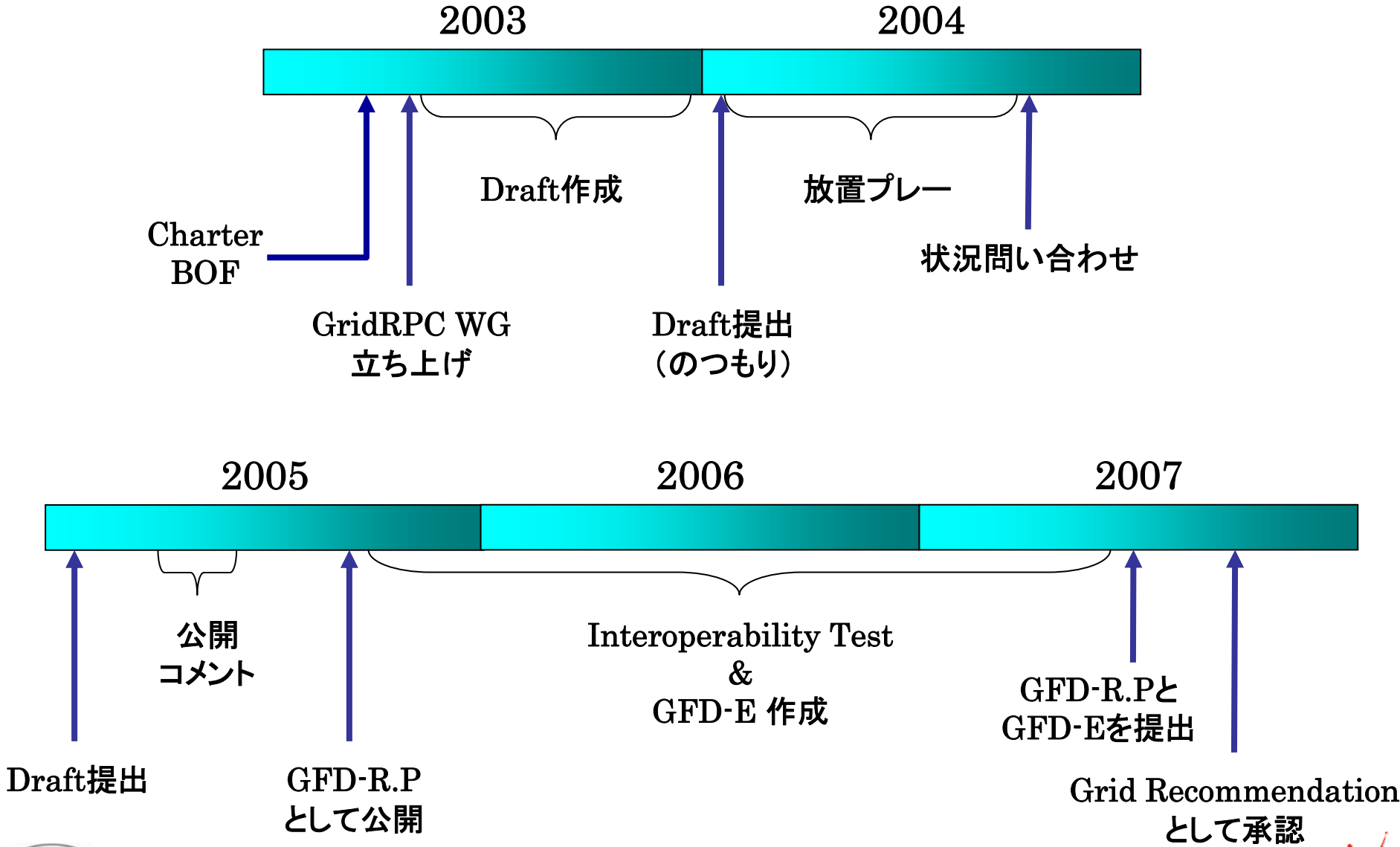
OGF Editor

GFSG

ドラフトを返却
ドラフト削除



History at a glance



終わりに： あがった感想

- 標準化プロセスがきちんと確立されていない印象を持った
 - ▶ ほとんどの処理はGridForge上で行われたが、いくつかはArea Directorへのメールなどを利用した。
 - ▶ 約9ヶ月の放置プレー
 - ▶ Interoperability Testドキュメントの見本がなかった
 - ◎ Interoperability Testとして何をして、ドキュメントをどうまとめればよいか良く分からなかった
 - ▶ なぜかGFD-EにETSIのreviewがついた
 - ▶ ETSIからのコメントにあった、用語の定義
 - ▶ 最終審査は「4ヶ月」となっているが、実際には4ヶ月弱だった。
 - ▶ Grid Recommendationとして承認されたが、別に何の連絡もなかった
- 既存の実装に基づいた仕様の策定と、End-User APIとMiddleware APIの分離が鍵となった
 - ▶ それでもInteroperability Testには1年以上かかった
- End-User APIだけでアプリが書けるか？
 - ▶ 少なくともNinf-Gでは、大規模アプリの実装は難しい
 - ▶ End-User APIではカバーされていない機能、APIが各実装にある
 - ◎ Array関数、Ninf-G Object、属性関連、など
- 標準APIの策定は、プログラミングモデル普及の第一歩
 - ▶ 参照実装はすでに多くのUser Baseを持ち、実績があるが、標準化によりGridRPCというプログラミングモデルにお墨付きがついた。
- キモはモチベーションと実績
- とにかくあがって良かった
 - ▶ があ〜っとやれば2年ちょっとでできたのかもしれないが、とにかく最終標準までたどり着いてよかった。