GFD-R.052
カテゴリー：推奨
GridRPC ワーキンググループ

H. Nakada, AIST
S. Matsuoka, Tokyo Institute of Tech.
K. Seymour, Univ. of Tenn., Knoxville
J. Dongarra, Univ. of Tenn., Knoxville
C. Lee, The Aerospace Corp.
H. Casanova, UCSD, SDSC

2004 年 9 月 23 日
2005 年 2 月 17 日
2005 年 7 月 21 日

エンドユーザアプリケーション向け GridRPC モデルおよび API

本文書の位置づけ

本文書では、グリッドにより可能になったリモートプロシージャコールに対して提案されているモデルと API について、推奨すべき点をグリッドに携わる関係者に提示する。配布は自由である。

著作権情報

概要

本文書では、GridRPC すなわちグリッド環境でのリモートプロシージャコール（RPC）のメカニズムに関し、モデルと API を提示する。とくに、ミドルウエアではなくエンドユーザアプリケーションに焦点を当てる。すなわちここでは、エンドユーザにとって十分と思われる比較的簡単な GridRPC モデルと API を紹介することにし、ミドルウエアを構築する際に必要となるような複雑な特性や機能などはそこに含めない。グローバル・グリッド・フォーラムの推奨化過程文書として、本文書の目的は GridRPC のためのシンタックスとセマンティクスを明確かつ一義的に定義することにある。これにより、成長を続けるユーザ基盤が複数の実装を活用できるようになる。本文書を発行する動機は、グリッド計算の簡単な選択手段を示すことにあり、その理由としては次の 2 点が挙げられる。（1）RPC はすでに確立されている分散型計算パラダイムである。（2）ネットワーク対応のサービスに対し、増大しつつあるユーザ基盤が存在する。本文書はこの選択手段を示すことにより、

複数実装の開発を促進する狙いもある。

目次

1. はじめに

本文書の目的は、GridRPC すなわちグリッド環境でのリモートプロシージャコール（RPC）のメカニズムのためのシンタックスとセマンティクスを、明確かつ一義的に定義することにある。これによりグリッド計算の簡単な選択手段を示すことになる。とくに本文書では、ミドルウエアパッケージで必要になるような複雑な特性や機能を用いないエンドユーザアプリケーションに焦点を当てる。したがってミドルウエアに関する問題やネットワーク対応のサービスに関連する重要な問題を議論することは本文書の範囲外であり、チュートリアル的な情報を提供することも目的ではない。ただし本文書につながる関連研究に関して、参考文献や指針を示す目的で「関連する研究」の節を設けた。本文書で扱うモデルと API については、[15]でその準備段階を示していた。そのより詳しい記述は[16]

に与えられている。COBRA との比較は[21]に示されている。

2. 基本 GridRPC モデル

Registry	レジストリ
Client	クライアント
Service	サービス
handle	ハンドル
lookup	参照
register	登録
call	呼び出し
results	結果

図 1　基本 GridRPC モデル

図 1 に基本 GridRPC モデルを示した。そこに書かれた機能はきわめて基本的なものであり、他の多くのシステムでも見られるものである。あるサービスをレジストリに登録する。クライアントはレジストリにコンタクトをとり、必要なサービスを参照する。レジストリはハンドルをクライアントに返す。そしてクライアントはそのハンドルを使ってサービスをコールし、やがてサービスが結果を返す。

本文書で採用する GridRPC 用語では、サービスハンドルは 1 つの関数ハンドルであり、簡単でフラットな関数名の文字列から特定のサーバにあるその関数のインスタンスへのマッピングを示すものである。関数ハンドルをイニシャライズすることで関数からサーバへの特定のマッピングが確立すると、この関数ハンドルを使うすべての RPC コールがそのバインディングで指定されたサーバ上で実行される。セッション ID は、API の中で使用することで、以前サブミットされたノンブロッキングコールのステータスの取得、コールが終了するまでの待機、コールのキャンセル、コールのエラーコードの確認、などの作業をユーザが行えるようにするものである。

3. 本文書の範囲

この簡単でありふれたモデルは、しかし複数の基本的な問題を提示している。それらの問題を同時に扱うことは明らかに不可能である。したがって本文書が定義するものとしないものとを明確にしておくことにする。

3.1 範囲内

本文書では、エンドユーザアプリケーションに対する API を理解し、活用するために必要な API の定義と最低限のプログラミングモデルだけに焦点を当てる。より具体的に言えば、クライアントとサーバの簡単なやりとりだけに目を向ける。というのも、これで使用シナリオのほとんどに対処できるからである。

3.2 範囲外

以下のトピックスは非常に重要ではあるものの、本文書の範囲外である。

・ミドルウエア

ミドルウエアは、たとえばコールのときまで知らされることのない特定の GridRPC における可変個数の引数など、エンドユーザ向けのコードには登場しないような状況を扱わなければならない。

・サービス検出

サービスの登録や参照を実際にいかに行うかという点については本文書では取り扱わない。こうした機能を可能にするために何らかのレジストリやグリッド情報サービスが使用できるものと仮定している。

・ノンフラットなサービス名

現行の API では、GridRPC サービスに対して単純な名前の文字列を想定している。GridRPC サービスを属性やメタデータのスキーマで記述、検出することは非常に便利なことではあるが、ここでは問題として取り上げない。

・一般的なワークフロー

グリッドワークフローを管理する一般的なメカニズムを定義することも、本文書の範囲外のことである。ただし、API を単純に拡張してワークフロー管理ツールを使えるようにすることは可能である。

・実装間の相互運用性

本文書では GridRPC の API に焦点を当てるため、クライアント、サーバ、レジストリ間の通信に使用するプロトコルについては言及しない。そのため相互運用性についても問題として取り上げない。

4. GridRPC API

まず、使用するデータの種類を定義することで GridRPC API を表現する。つぎにイニシャライゼーションコールとファイナライゼーションコール、関数ハンドル管理コール、関

数コールそのもの、制御コールと待機コールを提示する。それぞれのコールの定義には、コールが返すエラーコードの一覧も示す。

4.1 GridRPC のデータの種類

grpc_function_handle_t

このデータタイプの変数は、基礎にある GridRPC システムが選んだ特定のサーバに対してバインドされた特定のリモート関数を表している。この変数はユーザがアロケートする。関数ハンドルをイニシャライズした後、これを使って関連するリモート関数を必要な回数だけ起動することができる。関数ハンドルのライフタイムは、ユーザがハンドル・デストラクトコールを使ってその関数ハンドルを無効にするときに決まる。

grpc_sessionid_t

このデータタイプの変数は、特定のノンブロッキング GridRPC コールを表す。セッション ID を使うことで、コールの終了を検出、待機したり、コールのキャンセルやエラー状況確認などを行ったりすることができる。セッション ID もユーザがアロケートするが、そのライフタイムは自動的に決められる。ノンブロッキング GridRPC コールが作成されるとセッション ID は初期化される。そして（1）すべての返却引数を受け取り、（2）アプリケーションに対して待機関数が「コール終了」を返したときに、セッション ID は無効化あるいは破棄される。無効なセッション ID がいずれかの GridRPC コールに送られるとエラーが出る。

grpc_error_t

このデータタイプは、GridRPC 関数から返されるエラーやリターンのあらゆるステータスコードに使用される。

4.2 イニシャライゼーションおよびファイナライゼーションの関数

このイニシャライズおよびファイナライズの関数は、MPI のイニシャライズコールやファイナライズコールに似たものである。イニシャライゼーションの前やファイナライゼーションの後のクライアント GridRPC コールは無効になる。

grpc_error_t grpc_initialize(char*config_file_name)

この関数はコンフィグレーションファイルを読み込み、必要なモジュールをイニシャライズする。この関数が一度コールされると、その後のコールは GRPC_ALREADY_INITILIZED とともに返される。

エラーコード識別子	意味

GRPC_NO_ERROR	成功
GRPC_CONFIGFILE_NOT_FOUND	指定されたコンフィグレーションファイルが見つからない
GRPC_ CONFIGFILE_ERROR	コンフィグレーションファイルの解析、処理にエラー発生
GRPC_OTHER_ERROR_CODE	内部エラー検知
GRPC_ ALREADY_INITILIZED	関数が複数回コールされた

grpc_error_t grpc_finalize(void)

この関数は GridRPC が使用しているリソースを開放し、終了していない非同期性のコールをすべてキャンセルするものである。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_OTHER_ERROR_CODE	内部エラー検知

4.3 関数ハンドルのリモート管理関数

この関数ハンドルの管理関数グループは、関数ハンドルの生成と破棄を行うものである。

grpc_error_t grpc_function_handle_default(
grpc_function_handle_t*handle,
char*func_name)

これは、与えられた関数名に関連したデフォルトサーバを使って新しい関数ハンドルを作成するものである。このサーバは、デフォルトでは、あらかじめ決められたサーバか、あるいは GridRPC 実装のリソース検出メカニズムが動的に選ぶサーバである。サーバを選ぶプロセスは、実際のコールがハンドルに対して作られるまでは延期することもできる。サーバをいったん選ぶと、ハンドルを通じたすべてのコールはそのサーバへ送られなければならない。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_ SERVER_NOT_FOUND	GRPC クライアントがまだサーバを見つけ

	ることができない
GRPC_ FUNCTION_NOT_FOUND	GRPC クライアントがデフォルトのサーバ上で関数をまだ見つけることができない
GRPC_RPC_REFUSED	ハンドルの生成がサーバにより拒否
GRPC_OTHER_ERROR_CODE	内部エラー検知

grpc_error_t grpc_function_handle_init(
grpc_function_handle_t*handle,
char*server_name,
char*func_name)

これはユーザが明確に指定したサーバで新しい関数ハンドルを作成する。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_ SERVER_NOT_FOUND	GRPC クライアントが指定されたサーバをまだ見つけることができない
GRPC_RPC_REFUSED	ハンドルの生成がサーバにより拒否
GRPC_ FUNCTION_NOT_FOUND	GRPC クライアントが指定されたサーバ上で関数をまだ見つけることができない
GRPC_OTHER_ERROR_CODE	内部エラー検知

実装者へのアドバイス：

サーバ名の文字列の正確な形式は指定されていない。1 つの可能性として、host_name:port_number という形の文字列があげられる。別の可能性としては、リソース仕様言語で文字列を書くということである。
※実装者へのアドバイス終了。

grpc_error_t grpc_function_handle_destruct(grpc_function_handle_t*handle)

これは、指定された関数ハンドルに関係するあらゆる情報やリソースを開放するものである。また、そのハンドルにバインドした実行中のセッションがあれば、ハンドル自体を開放する前にそのセッションをキャンセルする。

エラーコード識別子	意味
GRPC_NO_ERROR	成功

GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_ INVALID_FUNCTION_HANDLE	ハンドルがポイントした関数ハンドルが有効でない
GRPC_OTHER_ERROR_CODE	内部エラー検知

grpc_error_t grpc_function_handle(
grpc_function_handle_t**handle,
grpc_sessionid_t sessionID)

これは、与えられたセッション ID に対応する関数ハンドル（すなわちその特定のノンブロッキングリクエストに対応する関数ハンドル）を返す。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_ INVALID_SESSION_ID	セッション ID が無効
GRPC_OTHER_ERROR_CODE	内部エラー検知

4.4 GridRPC コール関数

エンドユーザは 2 つの GridRPC コール関数を使うことができる。この 2 つのコールは似たものであるが、ブロッキング（同期）とノンブロッキング（非同期）の動作を引き起こすものである。ノンブロッキングの場合はセッション ID が返され、そのセッション ID は終了のテストのために後で使用される。

grpc_error_t grpc_call(grpc_function_handle_t*handle, <varargs>)

これは、可変個数の引数を持ったブロッキングリモートプロシージャコールを生成する。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_ SERVER_NOT_FOUND	GRPC クライアントが指定されたサーバをまだ見つけることができない
GRPC_ FUNCTION_NOT_FOUND	GRPC クライアントが指定されたサーバ上で関数をまだ見つけることができない

GRPC_ INVALID_FUNCTION_HANDLE	ハンドルがポイントした関数ハンドルが有効でない
GRPC_RPC_REFUSED	サーバが RPC 起動を拒否。おそらくはセキュリティ上の問題による
GRPC_COMMUNICATION_FAILED	サーバとの通信に何らかの不具合が発生
GRPC_OTHER_ERROR_CODE	内部エラー検知

grpc_error_t grpc_call_async(
grpc_function_handle_t*handle,
grpc_sessionid_t * sessionID,
<varargs>)

これは、可変個の引数を持ったノンブロッキングリモートプロシージャコールを生成する。終了の検知あるいは待機、コールのキャンセルやエラー状況確認に使用するため、セッション ID が返される。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_ SERVER_NOT_FOUND	GRPC クライアントが指定されたサーバをまだ見つけることができない
GRPC_ FUNCTION_NOT_FOUND	GRPC クライアントが指定されたサーバ上で関数をまだ見つけることができない
GRPC_ INVALID_FUNCTION_HANDLE	ハンドルがポイントした関数ハンドルが有効でない
GRPC_RPC_REFUSED	サーバが RPC 起動を拒否。おそらくはセキュリティ上の問題による
GRPC_COMMUNICATION_FAILED	サーバとの通信に何らかの不具合が発生
GRPC_OTHER_ERROR_CODE	内部エラー検知

本文書の GridRPC 推奨では、非同期コールが返された際に実装関係のどのオペレーションを終了することにするかという点については、定義していない。ただし、すべての非同期GridRPC コールは、入力引数バッファをユーザが修正することに問題がなくなってから、できるだけ早く返されなければならない。

理由：

リモートオペレーションが始まる前、そして結果が返される前に、できるだけ早く非同期 GridRPC コールを返すことで、GridRPC のユーザはリモートの計算と他のローカルの計算とを同時に行うことができる。非同期コールが返された後にユーザがバッファを修正できるようにすることで、可能な範囲でもっとも安全で単純なバッファ処理セマンティクスをユーザに対して提示できる。入力引数バッファを安全に修正できるようになる前に非同期コールを返すことによりパフォーマンスを改善することは可能だが、バッファ処理がより複雑かつ「危険」になるため、薦められない。以上が、現在の典型的な実装で行われている方法である。
※理由終了。

4.5 非同期 GridRPC 制御関数

以下の関数は、すでにサブミットされたノンブロッキングリクエストに対してのみ適用される。

grpc_error_t grpc_probe(grpc_sessionid_t sessionID)

このコールは、セッション ID（sessionID）で表現された非同期 GridRPC コールが完了したかどうかを確認するものである。完了していれば GRPC_NO_ERROR が返され、そうでなければ GRPC_NOT_COMPLETED が返される。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_ INVALID_SESSION_ID	sessionID が有効でない
GRPC_NOT_COMPLETED	コールが完了していない
GRPC_OTHER_ERROR_CODE	内部エラー検知

grpc_error_t grpc_probe_or(
grpc_sessionid_t *idArray,
size_t length,
grpc_sessionid_t *idPtr)

このコールは、idArray 内のセッション ID の配列により表現される非同期 GridRPC コールが完了したかどうかを調べるものである。コールが完了していれば関数の返却値は GRPC_NO_ERROR であり、*idPtr がポイントする grpc_sessionid_t には有効かつ完了したコールが 1 つだけ含まれることになる。コールが完了していなかった場合は、関数の返却値は GRPC_NONE_COMPLETED であり、*idPtr がポイントする grpc_sessionid_t は

定義されない。IdArray 内のセッション ID のうちいずれかが無効である場合はオペレーションが行われず、GRPC_INVALID_SESSION_ID のエラーが返される。ただしセッション ID の配列にエラーを出さずに完了したセッション ID が含まれていてもよい。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_ INVALID_SESSION_ID	idArray 内の SessionID が有効でない
GRPC_NONE_COMPLETED	idArray 内のコールが完了していない
GRPC_OTHER_ERROR_CODE	内部エラー検知

理由：

ユーザは通常、こうしたセッション ID の配列を埋め、それが 1 つずつ終了していくことをチェックする。そのため、そのような配列に完了したセッション ID が含まれることはよくあることである。穴だらけの配列によりパフォーマンスに問題が生じるのであれば、ユーザが自分で配列を埋めていくという選択肢もある。
※理由終了。

実装者へのアドバイス：

本文書では grpc_sessionid_t の実際の表現を規定しないが、内部のエラー確認に使用できるこの種のボイド変数あるいは無効な変数を表すのに、実装が何らかの値を用いることは可能である。たとえば grpc_probe_or()が GRPC_NONE_COMPLETED を返した場合、idPtr がポイントする grpc_sessionid_t はこの無効値に実際に設定されることがある。
※実装者へのアドバイス終了。

grpc_error_t grpc_cancel(grpc_sessionid_t sessionID)

これは、指定された非同期 GridRPC コールをキャンセルするものである。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_ INVALID_SESSION_ID	sessionID が有効でない
GRPC_OTHER_ERROR_CODE	内部エラー検知

grpc_error_t grpc_cancel_all(void)

これは、未決の非同期 GridRPC コールをすべてキャンセルするものである。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_OTHER_ERROR_CODE	内部エラー検知

理由：
「配列キャンセル」コールも考えられたが、エラーの処理が難しくなるため採用しなかった。
※理由終了。

4.6 同期 GridRPC 待機関数

次の 5 つの関数は、すでにサブミットされたノンブロッキングリクエストだけに適用される。これらのコールにより、セッション ID に対してポーリングを繰り返し行うのではなく、アプリケーションが完了に関する非決定性のセマンティクスを必要な際に基盤システムに対して示すことができる。

実装者へのアドバイス：

実装の観点からすれば、ポーリングで無駄になるサイクルを少なくするために、こうした情報を OS スケジューラに伝えることができる。
※実装者へのアドバイス終了。

grpc_error_t grpc_wait(grpc_sessionid_t sessionID)

これは、指定したノンブロッキングリクエストが完了するまでブロックを行うものである。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_ INVALID_SESSION_ID	sessionID が有効でない
GRPC_ COMMUNICATION_FAILED	サーバとの通信に何らかの不具合が発生
GRPC_SESSION_FAILED	指定したセッションに不具合が発生

GRPC_OTHER_ERROR_CODE	内部エラー検知

grpc_error_t grpc_wait_and(
grpc_sessionid_t *idArray,
size_t length)

これは、idArray 内の指定したノンブロッキングリクエストのすべてが完了するまでブロックを行うものである。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_ INVALID_SESSION_ID	idArray 内の 1 つあるいは複数の sessionID が有効でない
GRPC_SESSION_FAILED	1 つあるいは複数のセッションに不具合が生じた
GRPC_OTHER_ERROR_CODE	内部エラー検知

grpc_error_t grpc_wait_or(
grpc_sessionid_t *idArray,
size_t length,
grpc_sessionid_t *idPtr)

これは、idArray 内の指定したノンブロッキングリクエストのいずれかが完了するまでブロックを行うものである。リターンが問題なく行われると、idPtr が完了したリクエストに対してポイントする。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_ INVALID_SESSION_ID	idArray 内の 1 つあるいは複数の sessionID が有効でない
GRPC_SESSION_FAILED	idPtr がポイントしたセッション ID に不具合が生じた
GRPC_OTHER_ERROR_CODE	内部エラー検知

grpc_error_t grpc_wait_all(void)

これは、すでに発行されているノンブロッキングリクエストのすべてが完了するまでブロックを行うものである。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_SESSION_FAILED	1 つあるいは複数のセッションに不具合が生じた
GRPC_OTHER_ERROR_CODE	内部エラー検知

grpc_error_t grpc_wait_any(grpc_sessionid_t *idPtr)

これは、すでに発行されているノンブロッキングリクエストのいずれかが完了するまでブロックを行うものである。リターンが問題なく行われると、idPtr が完了したリクエストに対してポイントする。

エラーコード識別子	意味
GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_SESSION_FAILED	idPtr がポイントしたセッション ID に不具合が生じた
GRPC_OTHER_ERROR_CODE	内部エラー検知

注意：grpc_wait_or()および grpc_wait_any()については、複数のコールが完了した場合、完了したセッション ID のうちどれを idPtr がポイントするかは決められていない。つまり複数のコールが完了した場合、返されるセッション ID が最初に完了したコールのものであることは保証されておらず、また、それに続く待機コールが完了したセッション ID を特定の順序で返すことも保証されていない。

4.7 エラーコードとエラー報告関数

GridRPC に不具合が生じるとエラーコードが返される。grpc_error_t のタイプの変数とともに使用することのできるエラーコード識別子を表 1 に示した。

エラーコード識別子	意味

GRPC_NO_ERROR	成功
GRPC_NOT_INITIALIZED	GRPC クライアントがまだイニシャライズされていない
GRPC_CONFIGFILE_NOT_FOUND	指定されたコンフィグレーションファイルが見つからない
GRPC_ CONFIGFILE_ERROR	コンフィグレーションファイルの解析、処理にエラー発生
GRPC_ SERVER_NOT_FOUND	GRPC クライアントがサーバを見つけることができない
GRPC_ FUNCTION_NOT_FOUND	GRPC クライアントがデフォルトのサーバ上で関数をまだ見つけることができない
GRPC_ INVALID_FUNCTION_HANDLE	関数ハンドルが有効でない
GRPC_ INVALID_SESSION_ID	SessionID が有効でない
GRPC_RPC_REFUSED	サーバが RPC 起動を拒否。おそらくはセキュリティ上の問題による
GRPC_COMMUNICATION_FAILED	サーバとの通信に何らかの不具合が発生
GRPC_SESSION_FAILED	指定したセッションに不具合が発生
GRPC_NOT_COMPLETED	コールが完了していない
GRPC_NONE_COMPLETED	コールが１つも完了していない
GRPC_OTHER_ERROR_CODE	内部エラー検知
GRPC_UNKNOWN_ERROR_CODE	未知のエラーコードのためにエラー記述文字列がリクエストされた
GRPC_ ALREADY_INITILIZED	関数が複数回コールされた
GRPC_LAST_ERROR_CODE	最も高い数値エラーコード。バインドされたエラーコードに用いられ、実際のエラーを表すものではない

表 1　GridRPC のエラーコード

これらのエラーコードには次の関係がある。

0=GRPC_NO_ERROR < GRPC_... < GRPC_LAST_ERROR_CODE

これは、実装には関係なく、エラーコードを整数を使って順序づける便利なものである。

すでにサブミットされているリクエストのエラーコードを確認する機能は以上で与えられた。以下に示したエラー報告関数は、エラーコードと人間にも読めるエラーの記述を与えるものである。これらのエラーの記述は、エラーの実際の原因についてより詳しい情報を提供してくれる。

char *grpc_error_string(grpc_error_t error_code)

これは、GridRPC エラーコードが与えられると、それに対するエラー記述文字列を返すものである。エラーコードが何らかの理由で不明の場合は、GRPC_UNKNOWN_ERROR_CODE の文字列が返される。

grpc_error_t grpc_get_error (grpc_sessionid_t sessionID)

これは、与えられたノンブロッキングリクエストに関連したエラーコードを返すものである。

grpc_error_t grpc_get_failed_sessionid (grpc_sessionid_t *idPtr)

これは、最新の GRPC_SESSION_FAILED エラーに関連したセッション ID を返すものである。セッション ID のセットを扱うコールのために不具合が生じた特定のセッション ID に関し、さらなるエラー情報を提供するものであり、grpc_wait_all()_、 grpc_wait_and() などを含む。不具合の生じたセッションが 2 つ以上ある場合は、この関数はセッション ID を 1 つずつ順番に返す。不具合の生じたすべてのセッションに対して見落としがないことを確認するには、ユーザは GRPC_SESSIONID_VOID が返されるまでこの関数を繰り返し使用する必要がある。

理由：

GridRPC のエラーコードは、MPI 標準のエラークラスに類似させる意図がある。すなわち、これらは GridRPC API に固有の種類のエラーであり、GridRPC のいかなる実装においても発生しうる。実装に固有のエラー情報は、関連するエラー記述文字列に含まれている。実装に固有のエラーには、GRPC_OTHER_ERROR_CODE のエラーコードが使用されることがある。

※理由終了。

5. 関連する研究

リモートプロシージャコール（RPC）の概念は、分散計算および分散システムにおいて長年の間広く使用されてきた[4]。RPC は、分散コンポーネントが明確に定義されたセマンティクスとコミュニケートできるようにする的確で簡単な抽象化を提供する。RPC の実装には多くの難しい問題があり、適切なアプリケーションプログラミングインタフェース（API）、ワイヤプロトコル、インタフェース記述言語（IDL）などの定義の問題もそこに含まれる。対応する実装を選ぼうとすると、柔軟性、ポータビリティ、性能の間のトレードオフに行き着く。

これまでの多くの作業は、シングルプロセッサに対する高性能 RPC メカニズムや、共有メモリ・マルチプロセッサのような強く結合した同一機種の並列計算機に対する高性能

RPC メカニズムの開発に重点を置いてきた[7, 3, 13, 2]。これらの作業による寄与としては、OS やハードウエアの低レベルの機能に直接マップする RPC メカニズムを提供することで、高いパフォーマンスを達成することにある（たとえば[5]にあるように既存のメッセージ伝達メカニズム上にビルドされた実装から脱け出ること）。これとは逆に、GridRPC では、広域ネットワーク上のゆるく結合した異機種混在型システムを対象とし、さまざまな問題や目標を扱ってきた。

現在の作業は、アドバンストプログラミングモデル研究グループから派生したものである[10]。このグループは、GridRPC など、数多くのプログラミングモデル[11,12]を調査、評価している。代表的な GridRPC システムが、NetSolve [6,20]と Ninf [14,19]である。歴史的には、どちらのプロジェクトも同じ時期にスタートしており、実際に似たような特徴を持っている。RCS [1]や Punch（http://punch.purdue.edu）など、関連する多くの実験システムがある。これらのシステムは、グリッドのユーザが簡単にリクエストを机上からリモートアプリケーションサーバに送ることができる方法を提供するものである。GridRPC は、こうした過去の努力を 1 つにまとめようとしている。

この作業は、XML-RPC（http://www.xml-rpc.com）および SOAP [18]にも関係している。これらは HTTP を使って、入力パラメタを記述する XML フラグメントをパスし、RPC コールの間に出力結果を取得する。科学計算においては、RPC コールへのパラメタは数値データの巨大な配列になることがよくある（たとえば倍精度の行列など）。[9]は、XML エンコーディングを使用すると、この種のデータが問題になることを明らかにした（たとえば浮動小数点の精度の欠如や、エンコーディングやデコーディングのコストなど）。しかし最近の研究では[17]、OGSA [8]などのウェブサービスに基づいた今度のグリッドソフトウエア上に、GridRPC が効果的にビルドされることがわかっている。

6. セキュリティに関して

GridRPC のセキュリティの問題は、実装には無関係であり、本文書では API におけるセキュリティの問題をとりわけ取り上げるようなことはしない。参考のため、Ninf-G および NetSolve のセキュリティメカニズムを本節で解説する。Ninf-G のセキュリティインフラストラクチャは、公開キー暗号化、X.509 認証、セキュアソケットレイヤ（SSL）通信プロトコルに基づいた GSI を基礎にしている。これはすなわち、すべてのコンポーネントが、適切に保護されているだけでなく、GridFTP サーバのような他の Globus コンポーネントをシームレスかつ安全に使うことができることを意味する。NetSolve の現在のセキュリティは、NetSolve サーバへのアクセスを許可したり禁止したりするために使用するアクセス制御のリストを生成する機能に基礎を置いている。NetSolve は認証に Kerberos V5 サービスを使用している。Kerberos を使って NetSolve を拡張することで、計算リソースへのアクセス制御に使用する信頼性の高いメカニズムを提供できる。現時点で、NetSolve の Kerberos 版は、NetSolve のクライアント、サーバ、エージェントの間で交換しているデー

タの暗号化は行っておらず、データストリームを完全には保護していない。

著者連絡先

中田秀基
産業技術総合研究所
hide-nakada@aist.go.jp

松岡聡
東京工業大学、国立情報学研究所
matsu@is.titech.ac.jp

以下 4 名の著者は原文のまま

知的財産権について

本文書に記載された技術の実装や使用に関連すると考えられるいかなる知的財産権およびその他の権利についても、GGFは、その有効性や範囲に関して特定の立場をとるものではない。また、こうした権利下にあるいかなるライセンスについても、それが使用できるあるいは私用できない範囲について、特定の立場をとるものではない。さらに、これらのいかなる権利についても特定する努力をGGFが行うものではない。出版やライセンス保証のために用意した権利請求書、および、開発者や本仕様書の使用者が所有権を使用する上での一般的なライセンスや許可を取得する作業の結果については、GGF事務局から入手可能である。

関係者の方々は、本文書で薦められていることを実践する上で必要なあらゆる著作権、特許、特許利用、その他の所有権に対して注意を向けるよう GGF は希望する。情報は GGF 委員長までお寄せいただければ幸いである。

グリッド提言（Grid Recommendations）を開発する目的で改変が必要ならば、その限りではない。この場合、GGFドキュメントプロセスで決められた著作権に対する手続きを遵守する必要がある。また、英語以外の言語に翻訳する際に改変が必要な場合も、その限りではない。

上に与えた制限付き許諾は永続的なものであり、GGFあるいはその後続組織または委嘱組織が無効にすることはない。

本文書とそこに含まれる情報は保証されたものではない。グローバル・グリッド・フォーラムは、ここにおける情報の使用がいかなる権利をも侵害していないこと、市販性、特定目的との適合性に関するいかなる黙示保証をも侵害していないことを含む（ただし必ずしもこれらに限定されない）明示あるいは暗示の保証を拒否するものである。

参考文献

以下省略