Part 2

IBM i 7.3 の新機能とそのインパクト

ティアンドトラストの小川誠氏と資料を読み解く

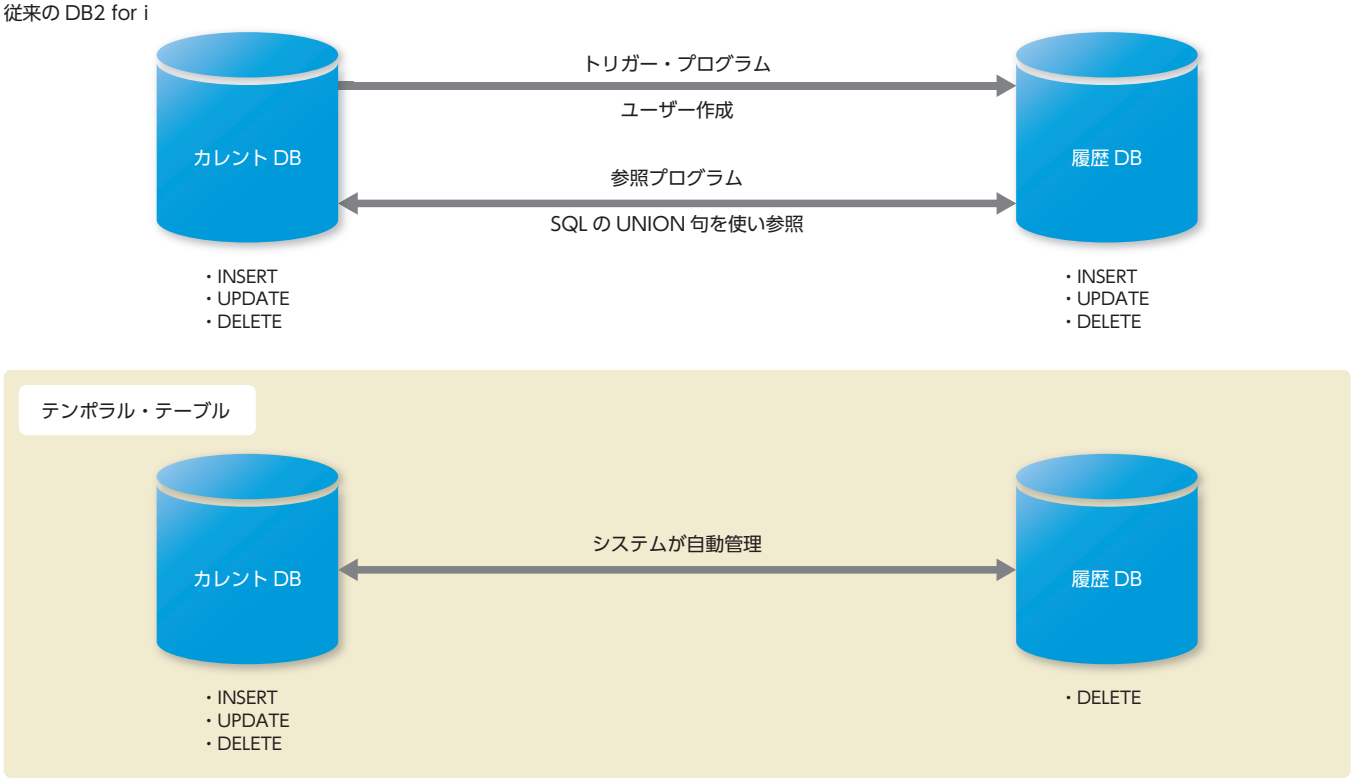
IBM i 7.3 の発表と同時に多数の製品資料が公開された。現段階ではテキストベースの資料が中心で、画面や仕組み図、動画資料などは少ないが、限られた資料を読み解くだけでも 7.3 のインパクトがうかがえる。IBM i やオープンソースを駆使した開発を幅広く手がけ、長年、IBM i のトレーニング講師も務めるティアンドトラストの小川誠氏に IBM i 7.3 の解説をお願いした。

GitとEclipse Orionのサポートが衝撃的

i Magazine（以下、i Mag）　IBM i 7.3の第一印象はどのようなものでしたか。

小川　ツイッターで「IBM i 7.3リリース」のツイートが流れてきたので、その時にざっと発表内容に目をとおしてみました。何よりもGitとEclipse Orionのサポートが衝撃的でしたね。後になって、いつもより詳しく調べてみましたが、今回はDB2とオープンソース関連の追加･拡張が目覚ましく、オープン系技術者をどうIBM iの世界に誘い入れるか､IBMの“本気度”を強く感じました。そのあたりが一気に進んだ印象です。

図表1　テンポラル・テーブルの仕組み



i Mag　オープン系への対応は、これまでのバージョンにもありましたが、それとは違う印象ですか。

小川　オープン系への対応はこれまで、レガシーのいいものを残しつつアドオンする形でなされてきたと思います。そのため、オープン系技術者がIBM iで開発や運用をするには、IBM iの基本事項を知らないと話にならなかった。そこを今回の新バージョンでは振り切って、オープン系の技術だけでもIBM iを扱える道が拓けてきた。その方向に、舵が一気に切られた感じです。

テンポラル・テーブル

履歴テーブルを自動で保守
日時・期間を指定して過去のデータを参照できる

i Mag　このテンポラル・テーブル（Temporal Table）　は、IBMの資料を見ると「テンポラル・サポート」や「システム期間テンポラル表」「システム時間テンポラル表」という訳語も当てられていますね。

小川 誠氏
ティアンドトラスト株式会社
取締役

小川　テンポラル・テーブルはSQL2011で搭載された機能ですが、「システム期間テンポラル表」ではわかりにくいですね。DB2（UDB）ではV10で、Oracle（V12）やマイクロソフトのSQL Server（2016）でも対応済みです。

i Mag　「ユーザー待望の」というようなコピーが目に付きます。どういう機能ですか。

小川　DB2のテーブルには通常、現時点の1次情報しか入っていませんが、たとえば5年前のある時点のデータを知りたい時、そのレコードを記録しておいてくれるのがテンポラル・テーブルの機能です（**図表1**）。

図表2　テンポラル・テーブルと履歴テーブル

テンポラル・テーブル

			監査例		システムが保守	
コード	商品	単価	ユーザー	I/U/D	開始列	終了列
A001	ピアノ	￥800,000			0001-01-01	9999-12-31
A002	ギター	￥250,000			0001-01-01	9999-12-31
A003	トランペット	￥150,000			0001-01-01	9999-12-31
A004	ベース	￥150,000			0001-01-01	9999-12-31

履歴テーブル

			監査例		システムが保守	
コード	商品	単価	ユーザー	I/U/D	開始列	終了列

図表3　USER1がレコードを挿入

テンポラル・テーブル

			監査例		システムが保守	
コード	商品	単価	ユーザー	I/U/D	開始列	終了列
A001	ピアノ	￥800,000			0001-01-01	9999-12-31
A002	ギター	￥250,000			0001-01-01	9999-12-31
A003	トランペット	￥150,000			0001-01-01	9999-12-31
A004	ベース	￥150,000			0001-01-01	9999-12-31
A005	ドラム	￥750,000	USER1	I	2016-03-25	9999-12-31

履歴テーブル

			監査例		システムが保守	
コード	商品	単価	ユーザー	I/U/D	開始列	終了列

図表2を見てください。左側のテーブル（テンポラル・テーブル）には、コード、商品、単価が入っています。このテーブルを、「テンポラル・テーブル」として使うために「開始列」と「終了列」をSQLで追加します（実際には、これ以外に必須のフィールドがありますが、わかりやすくするために省略しています）。開始列の最小値は「0001-01-01」、終了列の最大値は「9999-12-31」で、本来は実際のタイムスタンプから時・分・秒・ナノ秒が取得されます。**図表2**は現在生きているレコードなので、開始列は「0001-01-01」、終了列は「9999-12-31」になっています。そして右側の「履歴テーブル」は、テンポラル・テーブルに対して挿入・更新・削除が何もなされていないので、ブランクの状態になっています。

この**図表2**のテーブルにUSER1がレコードを挿入すると、左側のテンポラル・テーブルにレコードが挿入され、開始列はレコードを挿入した時点の「2016-03-25」となり、監査列（後述）のユーザーは「USER1」、挿入（I）・更新（U）・削除（D）

図表4 USER2がレコードを更新

テンポラル・テーブル

			監査例		システムが保守	
コード	商品	単価	ユーザー	I/U/D	開始列	終了列
A001	ピアノ	¥820,000	USER2	U	2016-03-28	9999-12-31
A002	ギター	¥250,000			0001-01-01	9999-12-31
A003	トランペット	¥160,000	USER2	U	2016-03-28	9999-12-31
A004	ベース	¥150,000			0001-01-01	9999-12-31
A005	ドラム	¥750,000	USER1	I	2016-03-25	9999-12-31

履歴テーブル

			監査例		システムが保守	
コード	商品	単価	ユーザー	I/U/D	開始列	終了列
A001	ピアノ	¥800,000			0001-01-01	2016-03-28
A003	トランペット	¥150,000			0001-01-01	2016-03-28

図表5 USER3がレコードを削除

テンポラル・テーブル

			監査例		システムが保守	
コード	商品	単価	ユーザー	I/U/D	開始列	終了列
A001	ピアノ	¥820,000			2016-03-28	9999-12-31
A002	ギター	¥250,000			0001-01-01	9999-12-31
~~A003~~	~~トランペット~~	~~¥160,000~~	~~USER2~~	~~U~~	~~2016-03-28~~	~~9999-12-31~~
A004	ベース	¥150,000			0001-01-01	9999-12-31
A005	ドラム	¥750,000	USER1	I	2016-03-25	9999-12-31

履歴テーブル

			監査例		システムが保守		
コード	商品	単価	ユーザー	I/U/D	開始列	終了列	
A001	ピアノ	¥800,000			0001-01-01	2016-03-28	
A003	トランペット	¥150,000			0001-01-01	2016-03-28	
A003	トランペット	¥160,000	USER2	U	2016-03-28	2016-04-10	
A003	トランペット	¥160,000	USER3	D	2016-03-28	2016-04-10	※

※＝ON DELETE AND ROW 指定時のみ追加（監査目的）

図表6 2016-03-27時点のレコードを照会（★対象レコード）

テンポラル・テーブル

				監査例		システムが保守	
	コード	商品	単価	ユーザー	I/U/D	開始列	終了列
	A001	ピアノ	¥820,000			~~2016-03-28~~	~~9999-12-31~~
★	A002	ギター	¥250,000			0001-01-01	9999-12-31
	~~A003~~	~~トランペット~~	~~¥160,000~~	~~USER2~~	~~U~~	~~2016-03-28~~	~~9999-12-31~~
★	A004	ベース	¥150,000			0001-01-01	9999-12-31
★	A005	ドラム	¥750,000	USER1	I	2016-03-25	9999-12-31

履歴テーブル

				監査例		システムが保守		
	コード	商品	単価	ユーザー	I/U/D	開始列	終了列	
★	A001	ピアノ	¥800,000			0001-01-01	2016-03-28	
★	A003	トランペット	¥150,000			0001-01-01	2016-03-28	
	A003	トランペット	¥160,000	USER2	U	2016-03-28	2016-04-10	
	A003	トランペット	¥160,000	USER3	D	2016-03-28	2016-04-10	※

※＝ON DELETE AND ROW 指定時のみ追加（監査目的）

を示すカラムには「I」が記載されます。終了列は、終了していないので「9999-12-31」のままです（図表3）。

次に、USER2が「ピアノ」と「トランペット」の単価を変更すると、テンポラル・テーブルの単価欄の値が変更され、開始列は変更した日時、監査列には「USER2」と「U」が記載されます。そして履歴テーブルには、変更前のレコードが取り込まれ、終了列のところが、変更された日の「2016-03-28」に変わります（図表4）。

次に、USER3が「トランペット」のレコードを削除すると、テンポラル・テーブルからそのレコードが削除され、削除されたレコードは履歴テーブルに引き継がれて終了列が削除日に変わります。指定がある場合は、監査列に「USER3」と「D」が記載されたレコードも追加されます（図表5）。

また、ある時点のレコードを照会する場合は、SQLのSELECT文を使ってその時点を指定すると、開始列と終了列の間で該当するレコードを選択します。図表6は、2016-03-27を検索条件に指定して照会した時の結果です。この要領で、ある期間のレコードを調べることが可能になります。

i Mag　こうした処理をテンポラル・テーブルを使わずに行うのは大変ですか。

小川　DB2と同じレイアウトのDBをもう1つ別に作成し、DB2でデータが挿入・更新・削除されたら、それをもう1つのDBに反映するトリガー・プログラムを書く必要があります。参照でも、両方のDBのレコードを見るためにSQLのUNIONの機能を使わなければなりません。それはちょっと面倒です。その面倒を避けるために、都度、クエリーを使って問い合わせるケースが多いのではないかと思います。これに対してテンポラル・テーブルでは、履歴テーブルをシステムが自動で保守し、参照もシステムが両方のテーブルを合わせてデータを見せてくれます。テンポラル・テーブルでは、初期設定（履歴テーブルの作成や2つのファイルの関連づけ）はユーザーですが、それ以降はシステムが自動でやってくれます（図表6）。

i Mag　業務アプリケーションでよく使われる処理ですね。

小川　過去のある時点のデータを知るための処理ですから、よく使います。過去のある時点のデータを知るということではバックアップなどの方法もありますが、バックアップだと1日1回程度がふつうですから、1日に複数回、変更された過程は見ることができません。テンポラル・テーブルだと、挿入・変更・削除のたびに記録されますから、正確な把握が可能です。当社の技術者にIBM i 7.3の資料を読ませたら、案の定、このテンポラル・テーブルに注目し、「これができるようになると、お客様からの問い合わせ対応が楽になる」とか「問い合わせ対応が不要になる」という話が出ていました。

i Mag　利用中のDB2 for iに適用するには、どうするのですか。

小川　現行のDB2へのフィールドの追加が必要で、それと同じレイアウトのファイルをもう1つ作成して、SQLコマンドを使って2つを連携させます。

i Mag　利用中のDB2とは別に履歴テーブルをもつファイルができるとなると、IBM i上のデータ量が膨らみますね。

小川　そうです。だから、すべてのDBを対象にするのではなく、どのファイルをテンポラル・テーブルで管理するかの選別が必要になります。マスター系のファイルの履歴管理には、特に有効な機能だと思います。

OLAP サポート

SQL によるデータ分析を提供
IBM i 上のデータ活用を拡大させる端緒

i Mag　OLAPのサポートもずいぶん強調されていますね。

小川　複数のレコードに順位を付けるRANK関数などはIBM i 6.1から利用できていますが、今回は一挙に多数のOLAP用関数がサポートされました。この中には、3つのレコードを足して、その合計値に対する各項目の百分率（%）を出す関数などもサポートされています。この計算式をRPGで書こうとしたら、けっこう大変です。

i Mag　資料を見ると、10種類以上の関数が追加されています。注目した関数はありますか。

小川　どれも面白そうなものばかりですが、2つの変数から相関関係を導き出す回帰分析用の関数は使い道が多そうで

す。広告費と売上高の相関分析から、ある広告投資額に対する売上高を想定するような使い方ですが、IBM iでこれをやるには、DB2のデータを別ファイルに移して、そのデータに対してOLAPツールを当てるしかないので、大きな効率化と言えます。Web Queryにも同様の回帰分析機能がありますが、今回はSQLだけでできるのがポイントです。

少し別の観点になりますが、私は今回のOLAP機能の追加を見て、あらためて統計学を勉強しようと思いました。IBM i上には過去からのデータが大量に蓄積されています。しかし、大半のユーザーは、その過去のデータをもっと有効活用できるはずです。今回追加されたテンポラル・テーブルの機能とOLAPサポートは、IBM iの世界でデータの活用・分析を大きく広げる、足がかりのような拡張ではないかと思えます。今回のOLAP機能を理解して統計学を習得すれば、ユーザーにさらに有効な提案ができるのではないか、とか、20年分のデータがあるのなら、そこから思わぬ情報を引き出せるのではないか、などと考えています。

監査列の生成

レコードの変更情報を自動的に記録
監査用として利用できる

i Mag　監査列の生成とはどのようなものですか。

小川　レコードに対する変更についての情報（変更のタイプ、ユーザー情報など）を追跡するためにシステムによって保守されるフィールドのことです。レコードの変更に関わるユーザー名やクライアントのIPアドレス、ジョブ名などの情報を、DBがレコード単位で自動的に生成します。先ほどの**図表5**を例にすると、「トランペット」のレコードをUSER3が2016-04-10に削除すると、右側の履歴テーブルにUSER3が2016-04-10に「D」したという証跡が記録されます。この例はテンポラル・テーブルの機能そのものですが、この「D」を設定する機能を提供するのが監査列の生成です。レコードの変更をシステムが自動的に記録するので、監査用に使えるというわけです。

i Mag　開発者を楽にする機能でもありますね。

小川　データベースの変更に対する履歴情報は、今まではプログラマーの責任において設定していましたが、ヒューマンエラーで記録されないことも残念ながらありました。システムが自動的に保守する監査列を使用すればその心配もなくなりますので、積極的に使用すべきですね。

セキュリティ・オーソリティ・コレクション

ユーザーのアクションを記録し
適切な権限設定を支援するセキュリティ機能

i Mag　セキュリティ・オーソリティ・コレクションはどういうものですか。

小川　ユーザーがあるプログラムをコールした時に、どの権限でアクセスしたのか、何のファイルに対してどのようなアクションを起こしたのかを記録するのがセキュリティ・オーソリティ・コレクションです。権限設定がユーザー（グループ）ごとにふつうになされている現状において、どうしてこんな機能が必要になるのかというと、たとえば、何の権限ももたないユーザーがシステムを利用する時、「*PUBLIC」でアクセスすると、システムによっては「*CHANGE」権限が割り振られてしまうことがあります。別のユーザーは「*USE」権限でアクセスし、権限どおりの利用しかできないのに対して、*PUBLIC権限だとそのようなことが起きる場合があるからです。

しかし、これらを精査するツールがなかったこともあり、これまでは見過ごされるのが一般的でしたが、「データ・セキュリティ違反が数多く報告されているので、多くのお客様はセキュリティ・ポリシーを検討しています」とIBM i 7.3の製品解説書にあるように、過剰な権限に起因するセキュリティ事案が多発しているのでしょう。

セキュリティ・オーソリティ・コレクションは、ユーザーのアクションに対してシステムがどう判断してアクセスを許可しているか、その記録を取ってくれるものです。そしてそのログを基に権限の過剰や対応策などを判断し、セキュリティ全体を向上させるのが目的です。

ログデータは、ユーザーごとに収集でき、システム提供のリポジトリに記録されます。ログの照会はSQLやNavigator for iで表示可能と解説書に明記されています。

DB2 Web Query for i 2.2

MySQL や PostgreSQL などをサポート
画面デザインの刷新に期待

i Mag　DB2 Web Query for iは2.1から2.2へとバージョンアップされ、MySQLやPostgreSQLなどの外部DBを新たにサポートしました。Type4のJDBCドライバーを使えば、Oracleなどへもアクセスできるとあります。

小川　これは、DB2 Web Query Standard Editionでのサポートで、廉価版のExpress Editionではサポートされないようです。MySQLやPostgreSQLを使っているオープン系の技術者は多いので、そういう人たちがIBM iの世界に入ってきた時に、IBM iと並んでMySQLやPostgreSQLも利用できるメリットは大きいだろうと思います。

i Mag　ほかに注目すべき点はありますか。

小川　現行のDB2 Web Query for i 2.1は画面デザインが古臭く、それがもう少し洗練されればもっと使われるだろうと感じていました。製品資料に「第一に特筆すべき変化は、サインオン画面とBIポータル画面がきれいにシンプルになったこと」とありますから、大いに期待しています。Webのどこを探しても、2.2の画面が見当たらないのが残念です。

それと、Web Queryの出力先にExcel（.xslx）とHTML5がサポートされています。ウインドウ・サイズに合わせて自動調整されるダッシュボードも提供されるので、使い勝手が大幅に改善されるものと思います。

IBM Navigator for i

多くの機能も追加

i Mag　Navigator for iはどのような拡張ですか。

小川　Navigator for iは、Webブラウザを使用した管理ツールです。7.3では、テンポラル・テーブルやセキュリティ・オーソリティ・コレクションに対応した拡張が含まれます。システム・モニターにはHTTPサーバーの要求受信率、応答送信率、要求処理時間などの情報のモニター機能も追加されました。また、時刻の同期を行うNTPの構成が可能になるなど、多くの拡張がなされています。

オープンソース対応

Eclipse Orion と Git により
IBM i 上でオープン系の開発スタイルを実現

i Mag　オープンソース関連ではさまざまな追加・拡張がありますが、小川さんが衝撃を受けたのはどういう点ですか。

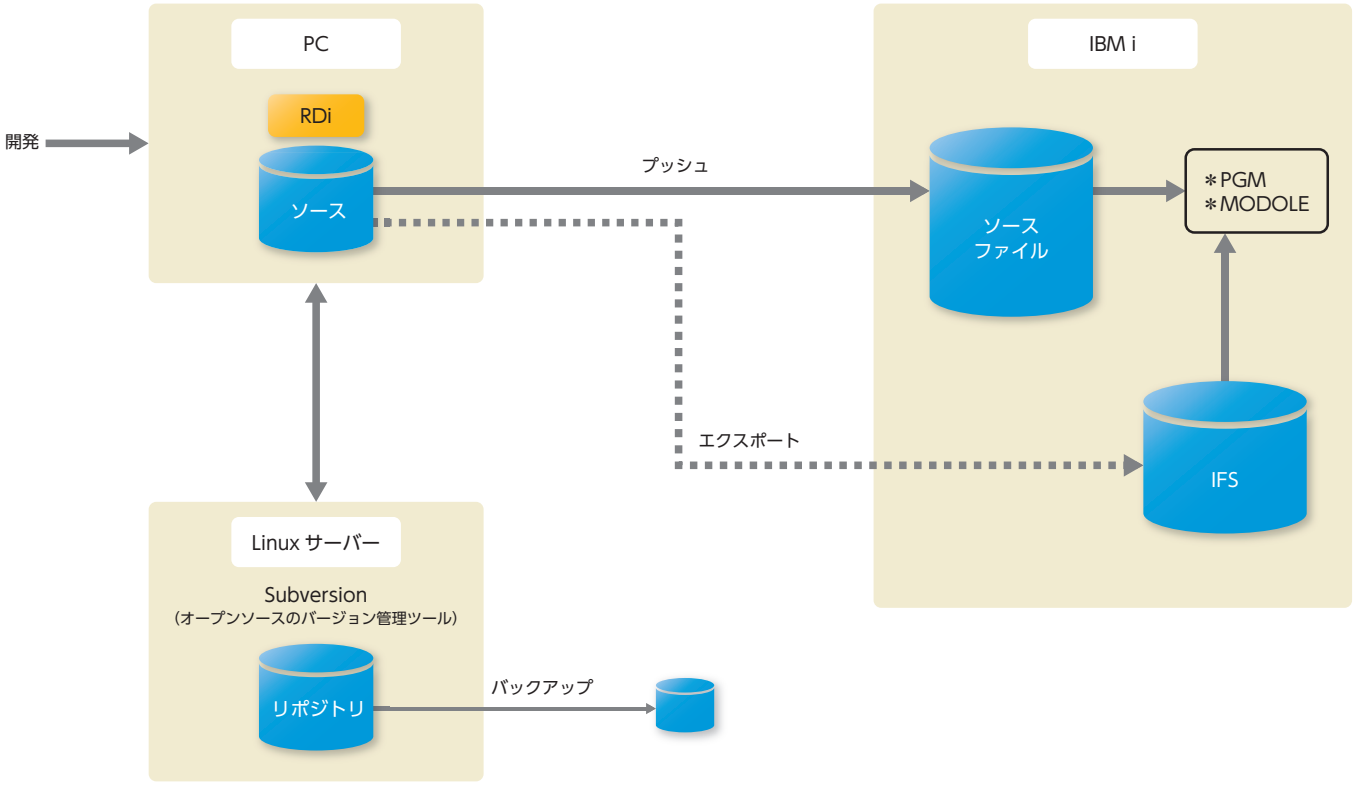
小川　その説明は、当社の開発環境を紹介するとわかりやすいと思います。当社ではRDiをPCにインストールして開発マシンとし、ソースのバージョン管理はオープンソースのSubversionを利用し、ソースをPCからIBM i上のソースファイルへプッシュする形で開発を行っています。Subversionは別途立てたLinuxサーバー上で稼働し、RDiのSubversionプラグインを使って連携します**（図表7）**。

今回のGitとEclipse Orionのサポートによってこうした開発環境がどう変化するかというと、まずWebブラウザベースのOrionを使うと、ソースはそのままIFSに格納され、バージョン管理はIBM i上のGitで行われるようになります**（図表8）**。つまり、オープン系のスタイルで一貫してIBM iプログラムを開発できるようになるわけです。この点が、私には衝撃的でした。

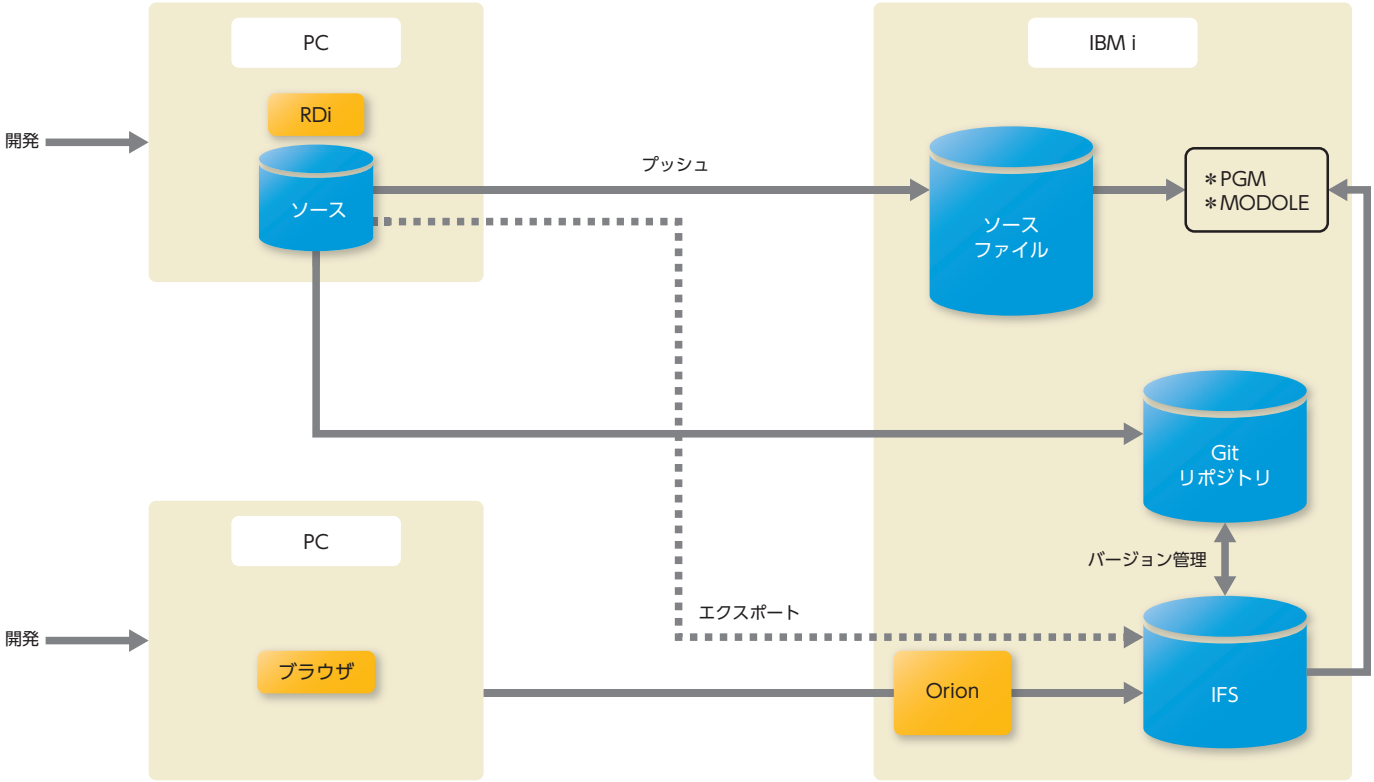
i Mag　RDiを使わなくてもいい開発環境の提供は驚きですね。

小川　資料によると、エディタとしてはRDiの代わりになりそうですが、システム開発にはデバッガなどのツールも必要で、Orionでどこまで可能なのかは、現時点ではわかっていません。その意味でRDiは今後も必要だと思います。Orionは、RDiのようなEclipseのプラグインをたくさんもつフル装備の

図表 7 ティアンドトラストの開発環境



図表 8 IBM i 7.3で可能になる開発環境



IDEではなく、Webベースの軽量な開発環境なので、PC上に広い画面の開発用エディタとソースをプッシュする機能さえあればいいという私のような技術者にとってはとても魅力的ですね。しかもOrionは、Node.jsやその他オープンソース言語のプラグインをもち、RPGの構文検査機能も備えています。

i Mag　Orionの利用が進みそうですね。

小川　IBM i以外のプラットフォームでは、フリーの開発環境を使うのがごくふつうなので、そう考えると、Orionという選択肢が浮上します。

それとOrionはEclipse環境なので、他の言語の開発者もIBM iでの開発をイメージしやすいというメリットがあります。Orionで開発したソースを、IFSファイル上に置けるので、JavaやPHPの開発者がLinux上のディレクトリにソースをputするのと同じ感覚で作れます。

RPGについては、フリーフォーム化以前はF仕様書やD仕様書などは等幅フォントを使わないとRDiなどのエディタでは桁位置がそろいませんでした。FF RPGになると桁位置を気にしなくてもいいので、フォントが自由に選べるようになります。それにより古臭さがなくなり、他の言語の開発者にとっても違和感がかなりなくなりますね。

i Mag　GitはクラウドではなくIBM i上に置かれるのですね。

小川　ライセンスとして登録されるので、おそらくIBM i上に配置されると思います。もちろん、ネイティブのRPGソースと、Node.jsやPHPなどのオープン系ソースの両方のバージョンを管理できますし、IBM i上にあるとIBM iと一緒にバックアップを取ることも可能になるでしょう。

i Mag　そのほかに注目しているものは何ですか。

小川　今回のオープンソース対応では、「Tools」でGCCも提供されています。GCCは、Linux環境でごく一般的なコンパイラで、CやC++、Objective-Cなどに対応しています。IBM iにもCコンパイラがありますが、オープンソース絡みではIBM i上でコンパイルできずバイナリを用意する必要があったり、IBM iネイティブでは稼働せず仮想システムに限定されているものもありました。GCCがサポートされたので、IBM i上でオープン系ソースのコンパイルが可能になりました。

i Mag　お話をうかがうと、IBM iの伝統的な開発からオープンな世界へと大きく踏み出した感じですね。

小川　本当にそう思います。これまでの新バージョンでは新機能がたくさん追加されてきましたが、今回は見た目と入り口が大きく広がった印象です。こうなると、今までのRPGのコーディング規約をまっさらにして、オープンな開発手法に則った規約を作っていく必要があると思います。IBM iのプログラム開発の方法が大きく変わりつつあると思えますね。