特集

より速く、より軽く TRON版 最新組込みデータベース

組込み機器にデータベースの導入が急ピッチで進んでいる。データベースといえば、大規模なデータを管理するエンタープライズ系システムでは以前から使われているが、リソースの少ない組込み機器ではあまり縁がなかったもの。それが今、ミドルウェアとして急速な広がりを見せているのは、組込み機器の高機能化やデータの大容量化など多くの背景がある。

そこで本特集では、T-KernelやITRONなどTRON系のOSに対応した商用の組込みデータベース製品および関連製品の最新動向を一挙に紹介する。

ニーズに合わせて普及が進む

組込みデータベース入門

石川 太一
株式会社日立ソリューションズ

00020A0F ucode

ここ数年、組込みデータベースを搭載した機器が増えている。たとえば携帯電話やミニコンポ、カーナビゲーションシステム、デジタルTVやSTB（セットトップボックス）、デジタルカメラなど多数ある。これらの機器にはなぜ、データベースを搭載しなければならなかったのか？

ビジネスモデルの変化がデータベースの普及を拡大

機器メーカーが組込みデータベースを搭載した背景のひとつは、機器を取り巻くビジネスモデルの変化である。

たとえば携帯電話では、パケット定額制度を前提としたコンテンツ配信サービスが増えている。着信メロディ配信サービスの場合、携帯電話に格納される楽曲コンテンツ数は最大数万曲にものぼる。そのため、利用者の操作性を向上するには、端末内の着信メロディを自動的に分類したり、すばやく探せるアプリケーションが必要になってきた。

カーナビゲーションシステムにも、携帯電話を接続して地図データを随時更新できるサービスが登場した。そのため、HDDに格納された数千万件もの地図データの一部を、端末側で随時更新するアプリケーションの開発を迫られるようになった。そこで、機器メーカーは組込みデータベースの採用に踏み切ったのだ。

もうひとつの背景は、HDDやフラッシュメモリなどの記録メディアの単価が下がり、より大容量の記録メディアを機器に搭載できるようになったことである。

たとえばデジタルカメラでは、2Gバイト以下だった記録メディアの容量が、今や数十Gバイトに増え、多くの写真を一度に撮り溜められるようになった。撮った写真を自動的に分類したり検索したりする機能はもはや不可欠である。

デジタルTVやSTBの分野では、とにかくたくさん録画してから、自分の好きな時間に見たい番組を探して視聴する「タイムシフト」のスタイルが定着しつつある。HDDに録画される番組数は最大数千件～数万件にもなる。そうなると、録画した番組の自動分類や検索機能なしでは使いこなせない。こうした記録メディアの大容量化も、組込みデータベースの導入に拍車をかけている（図1）。

組込みDBで解決できる課題は？

ほとんどすべてのアプリケーションは、何らかのデータをファイルに格納したり、読み書きをしている。

管理するデータ件数が少なく、検索などの複雑なデータ処理機能が必要なければ、配列やキュー、スタックといったデータ構造を使って容易にプログラムを作れるだろう。データ件数が少なければ、高速処理をねらってすべてのデータを組込み機器のメモリに展開しても、メモリを大量に消費する心配はあまりない。

ところが、管理するデータ件数が増加の一途をたどり、それに伴い性能が低下してくると、双方向リストやバイナリ・ツリーといったやや複雑なデータ構造を利用した性能対策が必要になる。また、データ件数が増えると、メモリに展開することも難しい。

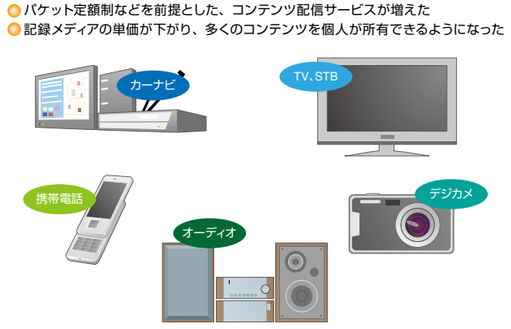

図1　機器を取り巻く環境の変化がデータベースの普及を拡大させている

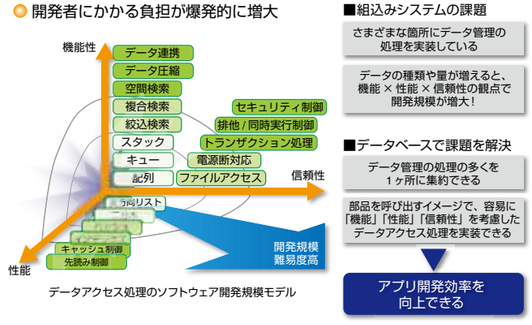

図2　組込みデータベースで「機能」「性能」「信頼性」の課題が解決できる

難しい。

機能面では、冒頭で紹介した携帯電話の着信メロディやTVの録画番組のように、大量のデータから目的のデータを検索する機能が求められる。単純なキーワード検索だけでなく、複数のキーワードを組み合わせた検索や分類も必要だろう。場合によっては、外部のサーバにあるデータベースの一部を検索したり、他の機器に分散配置したデータベースを検索するといったデータ連携機能が必要になることもある。

さらに、データ更新中の電源断対策のように信頼性を高める機能まで必要になってくると、プログラムのコードを大改造しなければならない。少しずつ機能を付け足して、複雑で読みにくくなったプログラムのコードに、機能をさらに加えることは簡単ではない。

このように、データ管理について「機能」「性能」「信頼性」のそれぞれの要求が同時並行して膨らんでいる（図2）。それをアプリケーションだけで対応したら、プログラムの開発ステップ数、つまりプログラム開発者にかかる負担は爆発的に増大してしまう。

そこで注目されたのが、組込みデータベースである。データ管理機能の大部分に組込みデータベースの機能を利用することによって、組込みプログラムの開発を容易にしながら、「機能」「性能」「信頼性」の要件を満たすことが期待されているのだ。

●検索画面の例

●システム構成の例

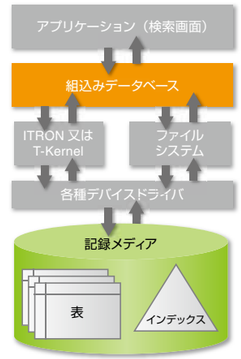

図3　周辺の目的地を検索するアプリケーション「歩行者ナビ」の例

組込みDBを使ってみよう

例として、周辺の目的地を検索するアプリケーション「歩行者ナビ」の開発を想定し、組込みデータベースを活用した開発工程の流れと、具体的な実装例を見てみよう（図3）。

データベースを利用することで開発効率を高められるだけでなく、データベースの導入が難しくないことを実感していただけるはずだ。

ここではITRONやT-Kernel上で動作するアプリケーションを想定しているが、開発工程やその具体的な開発手順は、どのOSやどのCPUを用いた場合でもほとんど同じである。これは、データベース層以下で、ターゲット機器側のOSやCPUの差異を吸収してくれるからだ。つまり、組込みデータベースを利用することで、ターゲット機器のOSの種類や、物理的なファイル格納位置などをあまり意識することなく、アプリケーションが開発できるのである。

では、歩行者ナビの開発手順を見ていこう。今回の開発手順は、日立ソリューションズの「Entier（エンティア）」[注1)]を使った場合の例であるが、リレーショナル型のデータベースであれば、どの製品を利用しても開発の流れはほとんど変わらない（表1）。

●データの種類を列挙する

まず、(1) のアプリケーションの要求仕様がある程度具体化した段階で、(2) のデータスキーマの設計、つまり表とインデックスを設計する。この工程では、アプリケーションが取り扱うデータの種類を列挙し、どの種類のデータをキーに、どの種類のデータを検索または更新するのかを設計する。

ここでは、「物件名称」「ヨミ」「電話番号」「住所」「郵便番号」「ジャンル」などのデータを扱うとしよう。この場合、表の構成は表2のようになる。また、目的地を検索する機能を実装したいので、「ジャンル」と「住所」をキーに「物件名称」を検索するためのインデックスを設定する。ここで設計した表およびインデックスは、リスト1のSQLを実行するだけで簡単に実装できる。

●問い合わせ手順を決める

次は (3) のSQL設計工程である。ここでは、データベースを検索または更新する単位で、データベースへの問い合わせ手順を決定していく。

たとえばデータベースの物件管理表に対して、「住所」に“大手町”というキーワードを含み、さらに「ジャンル」が“コンビニエンスストア”(ジャンルコードが“1”) の物件だけを絞り込み、その物件の「物件名称」と「住所」を表示したいとすると、SQL文はリスト2のようになる。

●データベースを構築する

それでは実際にデータベースを構築し、SQLを動かして検証してみよう。

組込みデータベース製品の多くには、PC上で動作するデータベースアクセスツールが付属している。ここではEntierのSDKに付属しているデータベースアクセスツール「Entier Control Manager」を使って手順を紹介しよう。

(5) のデータベースの構築工程は、大きく以下の手順で進められる。

①データベースを初期化する

初期化の際は、データベースを構成する物理的なファイルサイズなど、初期化パラメータを細かく指定できるが、今回はすべて既定値を使用する。基本的にはDB初期化ボタンをクリックするだけで、初期状態のデータベースファイルが作成できる (図4)。

②データスキーマを定義する

先ほど設計したデータスキーマ定義のSQL文 (リスト1) をそのまま図5のように実行する。その結果は左下のツリービューで簡単に確認できる。

③初期データをインポートする

初期データをインポートするには、インポートするCSV形式のファイルと、CSVデータを格納する表をデータインポート画面で指定するだけでよい (図6)。

以上のように、データベースの構築作業は難しくない。

●SQLを実行して検証する

最後に、設計工程で作成したSQLの動作を検証してみよう。「Entier Control Manager」で図7のようにSQLをそのまま実行すると、データベースの検索結果が表示される。

この例では、データベースの物件管理表に対して、「住所」に“大手町”というキーワードを含み、なおかつ「ジャンル」が“コンビニエンスストア”の物件だけ (ただし検証用に作成した架空のデータ) が表示された。

ここでは紹介できなかったが、(6) のプログラム作成工程のコーディング例や、アプリケーションの組込み方法は、日立ソリューションズのサイト[注2)]で詳しく紹介しているので、興味があれば確認していただきたい。

以上のように、データベースを利用すればデータ管理機能の大部分はプログラムを作成することなく、簡単に実装して動作を検証でき、ターゲット機器のOSやCPUの差異、物理的なデータ格納場所を意識する必要もないことがおわかりいただけたと思う。

表1 「歩行者ナビ」のデータベース設計と開発工程

工程		概要
(1)	アプリケーションの要求仕様定義	アプリケーションの要求仕様から、データベースアクセスが必要になる場面をすべて洗い出し、具体化する
(2)	データスキーマ定義	組み込みデータベースがアクセスする表と列、およびインデックスを定義する
(3)	ＳＱＬ設計	無駄のない、効率の良いＳＱＬ文を設計する
(4)	データベース設計	処理構成を考慮して、メモリとストレージの使い分けなどを設計する
(5)	データベース構築	(4) までに作ってきた設計情報をもとに、データベースを構築する
(6)	プログラムの作成	ＳＱＬを実行するプログラムを作成する
(7)	チューニング	各種統計情報を採取して、作成したプログラムに性能向上の余地がないか確認する

表2 物件管理表：歩行者ナビのデータスキーマ (表の構成) の例

列名称	データ型
物件名称	varchar(200)
ヨミ	varchar(200)
電話番号	pack(13,11)
住所	varchar(150)
郵便番号	integer
ジャンル	integer

リスト1 表およびインデックスを実装するSQLの例

```
●物件管理表の表定義文
CREATE TABLE 物件管理表
  (物件名称 varchar(200),
  ヨミ varchar(200),
  電話番号 pack(13,11),
  住所 varchar(150),
  郵便番号 integer,
  ジャンル integer);

●物件管理表に設定するインデックス定義文
CREATE INDEX IDX1 on 物件管理表(ジャンル);
CREATE INDEX IDX2 on 物件管理表(住所) TEXTSEARCH;
```

リスト2 大手町のコンビニエンスストアを検索するSQLの例

```
SELECT 物件名称, 住所 FROM 物件管理表
    WHERE CONTAINS (住所,'("大手町")=true  AND ジャンル = 1;
```

注1) Entierの体験版SDK (ソフトウェア開発環境) は、下記のURLから無償でダウンロードできる。
http://hitachisoft.jp/products/embedded/entier/
注2) Entierチュートリアル「4 サンプルプログラムを実行しよう」
http://hitachisoft.jp/products/embedded/entier/doc/tutorial.pdf

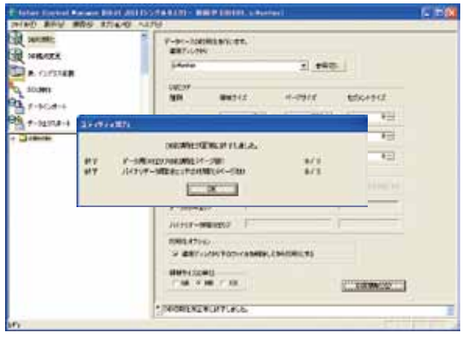

図4 データベースを初期化する画面例

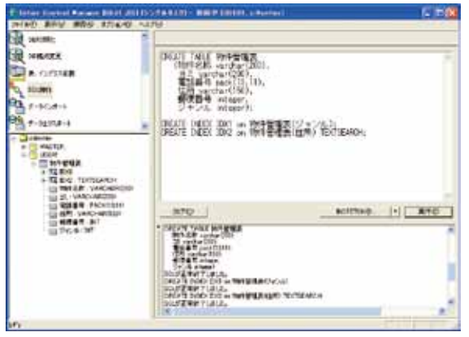

図5 データスキーマを定義する画面例

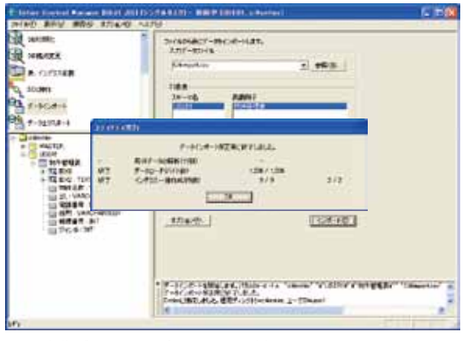

図6 初期データをインポートする画面例

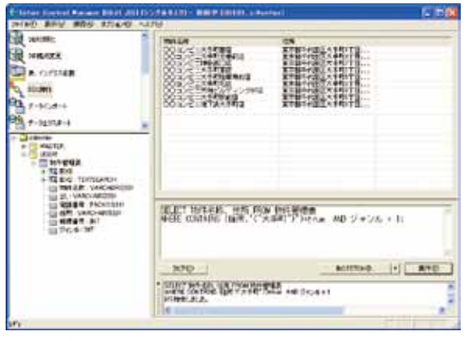

図7 設計したSQLの動作を検証する画面例

組込みアプリの多様性に対応

組込みシステムを取り巻く環境の変化により、アプリケーションの要求仕様は急速に多様化している。単純な機能であれば、どのデータベース製品を利用しても実装できるが、システム要件が複雑になると、実装や検証が困難になるケースもある。

2010年現在、ITRONやT-Kernel上で動作する組込みデータベース製品の選択肢は数多い。リレーショナル型のデータベースだけでも、たとえば機能は制限されるがメモリが少なくてすむもの、あるいはカーナビや携帯電話のアプリケーション要件に最適化したもの、サーバにあるデータベースとのデータ同期機能を備えたものなど、さまざまな特徴を備えた製品がある。

またリレーショナル型のデータベース以外にも、定型的なデータ管理に向く構造型データベースや、データ構造が変化するXMLをそのまま管理できるXMLデータベース、さらには各種データベースに不足した機能 (インデックス) を補完して高速化できるものまで、多様化するアプリケーション要件に応えてデータベースの種類は増えている。

データベースの活用を検討する際は、開発したいアプリケーションの要件に合わせ、適切なデータベース製品を選択してほしい。