吉原 泰介

株式会社ミガロ.

RAD事業部 技術支援課

[Delphi/400] Beacon技術によるIoT活用の第一歩

略歴
1978年3月26日生まれ
2001年 龍谷大学法学部卒業
2005年7月 株式会社ミガロ.入社
2005年7月 システム事業部配属
2007年4月 RAD事業部配属

現在の仕事内容
Delphi/400やJC/400の製品試験、および月100件に及ぶ問い合わせサポートやセミナー講師などを担当している。

1.はじめに

最近よく耳にするIoT（Internet of Things）とは、モノとインターネットをつなげて、データ収集、情報発信、自動的な運用動作などに活用する技術・考え方である。

PCやスマートデバイスであれば、もちろんそれ自体がネットワークを通じて情報を発信できるが、IoTではこれまでネットワークに接続していなかったモノこそが、重要な対象になる。【図1】

たとえば最近では、リストバンドで脈拍や移動距離などがネットワークで連携され、健康管理をスマートフォンで行えるようなIoTソリューションが増えてきた。

また車がネットワークにつながることで走行履歴、ガソリンの量、渋滞・災害情報などをリアルタイムに把握し、その状況に応じて最適なルートをナビゲーションするアプリケーションも研究されている（IoTによるカーナビの進化）。

IoTは定義としては非常に広いが、こうした技術や考え方が基盤となっている。そこでは、さまざまに電子化が進むなか、モノがどのように情報を発信するかが重要な鍵になる。

情報を発信する機器の1つに、Beaconがある。たとえば店舗に行くと、タイムセール情報や割引クーポンの情報が受信できるスマートフォンのアプリケーションが増えてきた。そうしたサービスの多くで、Beaconが使用されている。

IoTイコールBeaconというわけではないが、場所に特化したIoT技術として、最近とくに注目されている。

こうした状況を背景に、本稿ではIoTで活用されているBeaconの技術を題材とする。Beaconの基本的な情報から、IoTプログラムとして活用するため、Delphi/400による実装方法などについて説明する（本稿ではBeaconに対応したコンポーネントの使用が中心となるので、Delphi/400のバージョンは10 Seattleを対象にする）。

2. IoTとBeacon技術

2-1. Beaconとは

Beaconは、Bluetoothの信号を発信する機器である。その信号は数秒に1回、半径数十メートル範囲に発信される。ただし普通のBluetoothだとすぐに電力を使い果たしてしまうので、BeaconではBluetooth Low Energy（BLE）という、極力電力を使わない規格を使用している。【図2】

Beaconを使ったシステムでは、スマートフォンなどにインストールした専用アプリケーションでその信号を受信し、それをトリガーにして処理する（プログラムが動作する）仕組みとなっている。

Beaconを使う前提は、受け手が対応するアプリケーションをもっていて、BluetoothをONにするだけである。特別な操作は必要なく、信号に自動で反応するシンプルさがBeaconの魅力である。

次に、こうしたBeaconを使った活用

図1

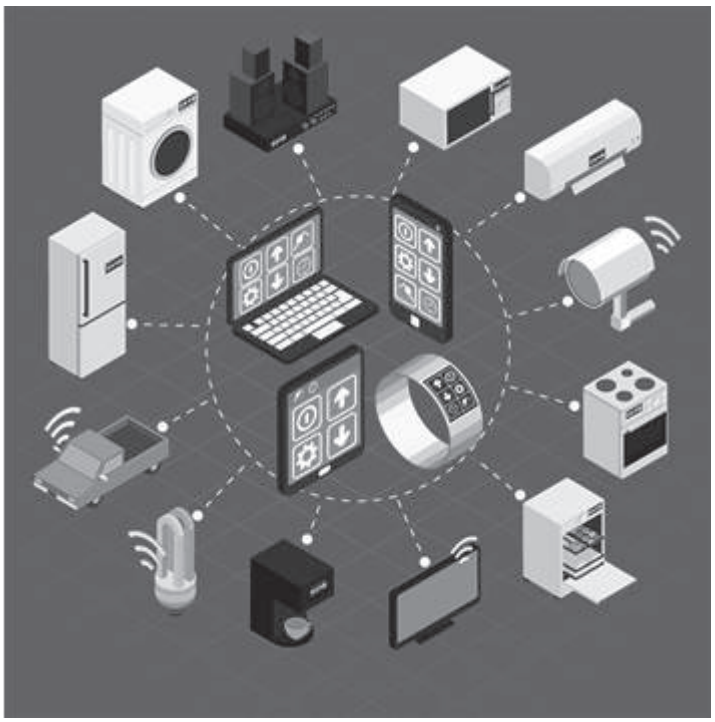


図2

Bluetooth™ 4.0 Low Energy

図3

出典：STOREBEACON iBeaconとは？

事例について考えてみる。

2-2. Beaconの活用事例

一般にBeaconを使う場合、店舗や工場内に信号を発信するBeacon機器を設置する。Beaconの活用用途としては、主に2つが考えられる。1つは情報の「通知サービス」、もう1つは施設内における高精度な「位置サービス」である。

たとえば店舗を訪れた顧客が専用アプリケーションをインストールしていれば、店舗に設置したBeacon機器の信号により、クーポンなどの特典を受信できるサービスが実現可能である。【図3】

Beacon機器から信号を受信したタイミングで、サーバーから情報を送ったり、受けたりできる。これが、「通知サービス」である。こうしたアプリケーションは、そのサーバーと連携できるので、BeaconとインターネットをIoTとしての活用につながる。【図4】

情報を受け取るだけではなく、たとえばオフィスでの活用例としては、社内にBeacon端末を設置し、従業員が出社すると自動的に出勤時間を打刻するといった勤怠管理の仕組みにも応用されている。

最近では、いつも買い置きする特定商品についてBeaconを利用することで、スマートフォンのアプリを経由して、自動的にショッピングサイトに自動注文できるといった仕組みも開発され始めている。

また、配置されたBeacon機器の信号をアプリケーションで受け取るということは、その機器の近くで行動していると推定できる。これを応用した技術が、「位置サービス」である。【図5】

位置サービスでは、後述するように、複数のBeaconで場所を特定していく必要があるが、これによって店舗のどの商品がある場所にいるか、どういった順番で場所を移動しているかなど、行動情報を蓄積してマーケティング分析にも利用されている。

屋内での位置サービスをビジネスに活用する例としては、大規模なショッピングモールでの道案内をはじめ、工場や倉庫など大きな施設内で作業工程上の動線としても利用されている。

また美術館や公共施設の展示物の前にBeacon機器を設置し、利用者が近付くと所持しているスマートフォンにその情報を通知するといった用途でも使われている。実際に宮崎県立西都原考古博物館では、Delphiで構築されたBeaconの館内案内システムが運用されている。

2-3. Beaconと類似する技術

Beaconのように信号を受信したり、位置情報を扱う技術はほかにも存在する。ここからは通知サービスと位置サービスを例にして、類似した技術と比較しながらBeaconの特徴を考えてみる。【図6】

まず通知サービスのように、近くで信号を受信することでアプリケーションを動作させる技術としては、NFC（Near Field Communication）が類似している。

NFCは、超近距離無線通信機能により通信する技術である。主に電子カードの決済などに使われており、技術としては似ているが、用途はBeaconと異なる。

BeaconとNFCの用途の違いとしては、距離と動作対象数が挙げられる。Beaconは十数メートルの範囲で機能するのに対して、NFCは接触に近い数センチの距離で機能する。そのため、Beaconはざっくりとした広範囲動作を得意とするが、NFCは近距離で精度の高い動作を得意とする。

また、Beaconは機器とアプリケーションが1対Nで動作するのに対して、NFCは1対1で動作させる。NFCが活用されているSuicaなど、交通カードを使った改札精算をイメージするとわかりやすい。

そのため、場所を起点とした一括情報サービスではBeacon、電子決済など個々の情報制御サービスではNFCが使われるといった違いがある。

また位置サービスの技術として広く活用されているものとして、GPSがある。「位置サービス＝GPS技術」と連想される場合も多い。しかしBeaconとGPSでは、得意とする分野が大きく異なる。

Beaconが屋内の精密な位置情報検知を得意とするのに対し、GPSは屋外での大まかな位置情報検知を得意とする。

またGPSは機器などの設備が不要な半面、建築物などで電波が精密に受信できないため、屋内での位置情報検知には活用できないことが多い。

一方Beaconは機器の配置を前提とするが、屋内で独自の位置を算出できるので、施設などでの精密な位置サービスに向いている。

次に、こうしたBeacon技術をDelphi/400により実装して活用する方法を検証していく。

3.Beaconを活用するプログラム開発

3-1. Beaconがもつ信号情報

まずBeaconをプログラムで扱ううえで、Beaconがどのような信号を発信しているかについて説明する。

Beacon機器が発信する信号の識別にはUUID、Major、Minorの情報が含まれている。これはBeacon機器に設定するもので（設定方法は機器によって異なる）、任意に指定できる。

一般には、下記のようなルールで設定することが多い。

- UUID：施設レベルで一意
- Major：フロア／エリアレベルで一意
- Minor：施設、フロア内での機器として一意

もちろん任意のルールで設定もできるが、こうした体系を組んだ指定のほうが、システムで組み込む際にどの施設のどのフロア（あるいはエリア）の機器かをわかりやすく管理できる。

3-2. Beaconで使用するコンポーネント

Delphi/400 10 SeattleではBeaconを使うために、TBluetoothLEやTBeaconというコンポーネントが用意されている。

TBeaconではIDなどを設定するだけで、対象のBeacon機器に反応して動作するアプリケーションを簡単に実装できる。

- TBluetoothLE コンポーネント

画面上にドラッグ＆ドロップするだけで、Beaconが使用するBluetoothLEに対応できる

- TBeacon コンポーネント

図4

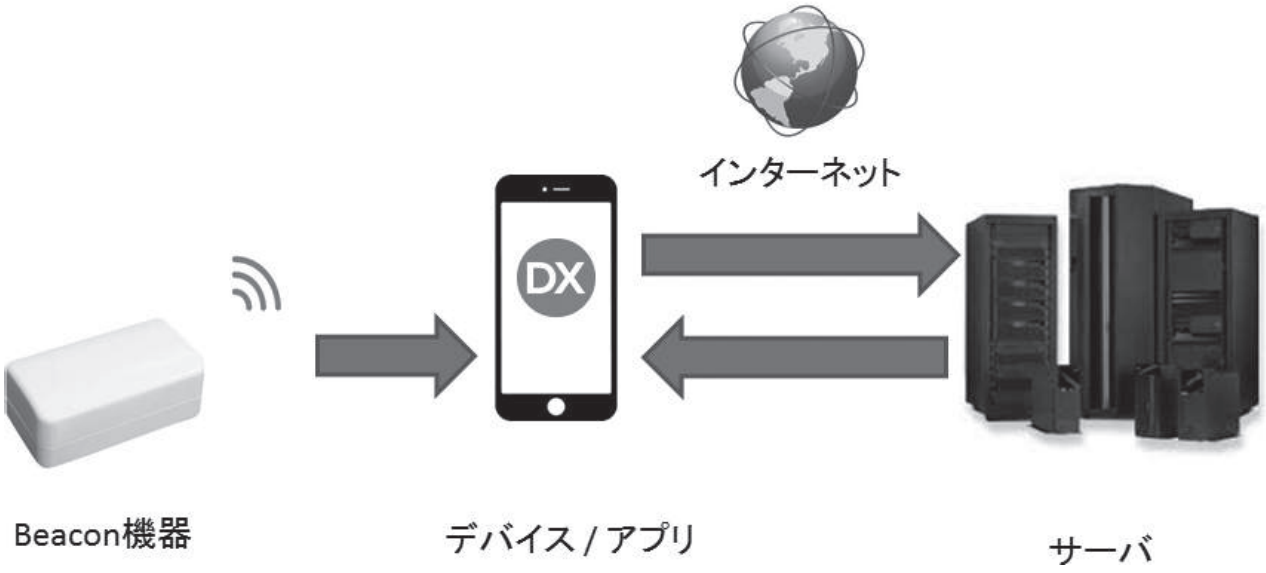
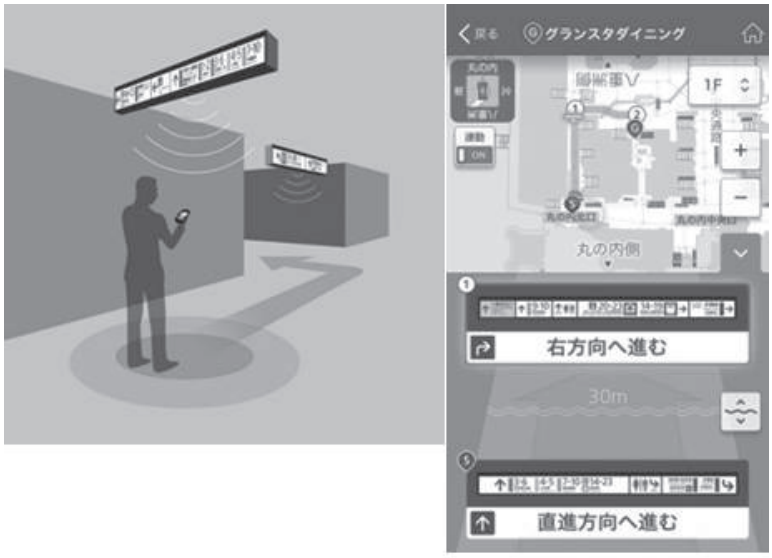


図5



出典：東日本旅客鉄道株式会社
実証実験「東京駅構内ナビ」

図6

Beaconの信号情報を感知・判別して、イベントを処理できる

3-3. Beaconを使ったプログラミング実装

Beaconを使ったプログラムを作成するにはコンポーネントの配置、コンポーネントの設定、そしてプログラミングによる処理作成が必要になる。ここでは、Beaconの信号に反応して広告画像を自動表示する簡単な通知サービス・アプリケーションを題材に、実装方法を説明していく。

まず、FireMonkeyで新規のマルチデバイスアプリケーションを作成し、画面フォーム上にTBeaconとTBluetoothLE、TImageコンポーネントを配置。TImageコンポーネントに、表示したい広告画像を設定しておく。【図7】

TBluetoothLEは貼り付けておくだけでBluetoothLEに対応できるので、特別な設定は必要ない。

TBeaconでは、簡単な設定とプログラミングを実装する。

まず設定内容として、対象とするBeacon機器のUUIDやMajor、Minorの値をプロパティに指定する（IDを機器に設定していない場合は、3-1.を参考）。TBeaconはアイテムとしてmonitorizedRegionsオブジェクトをもつので、このアイテムのMajor、Minor、UUIDプロパティで非常にわかりやすく設定できる。【図8】

IDのデフォルト値は、-1や0で設定される。Major、Minorの-1はすべてを対象にするので、別のエリアで同じUUID機器がある場合は注意が必要である。

またDataSnapサーバーと通信する場合はTSQLConnection、TDSProviderConnection、TClientDataSetなどのコンポーネントを配置・設定しておく（本稿ではDataSnapサーバーとの通信については割愛する）。

次に、Beacon機器の信号を受信した際のプログラム動作を実装する。TBeaconコンポーネントの基本的なイベントはOnEnter、OnExitである。このイベントで信号の受信成否を判定して、プログラミングが行える。

たとえばOnEnterイベントでTImageコンポーネントのVisibleをTrueにすれば、信号を受信して広告画像をアプリケーションで表示する。また同時にサーバーとのデータ連携が必要であれば、TClientDataSetを使ってサーバーへデータを登録するなど、IoTとしてシステムを連携できる。【ソース1】

信号を受信しなくなれば、OnExitイベントでVisibleをFalseにすることで広告画像を非表示にできる。【ソース2】

またアプリケーションの初期処理としてBeacon機能を有効にし、画像は非表示にしておく処理を、フォームのOnCreateイベントで作成しておく。【ソース3】

このプログラムをiOSやAndroidなどのスマートデバイスに対してコンパイルすると、アプリケーションは完成である。アプリケーションを実行して対象のBeacon信号を受信すると、広告画像が自動で表示される。【図9】

ここまで画像の表示・非表示の単純な実装方法を説明したが、同じイベントでサーバーに対して処理を行えば、Beacon機器の位置を起点にしてシステムを動かせる。

たとえば、Beacon機器を使った勤怠管理システムがこうした単純な仕組みで動作している。画像の表示の代わりに、受信時刻で入室データをDBへ登録しているのである。

同様に、工場の生産工程で作業場所に応じてチェック処理を自動的に起動したり、倉庫の商品棚管理では位置を自動登録するなど、場所を起点としたIoTシステムで活用されている。

また単純なBeacon信号の受信有無だけではなく、その信号電波の強弱によって機器までの近接度を判別することもできる。

近接度（ABeacon.Procimity）は、次の4段階で判定する。

- Immediate：0.5m未満
- Near：0.5m以上、1.5m以下
- Far：1.5mより遠い
- Away：距離判定不能

近接度をOnEnterイベントで判定する方法を【ソース4】に示すので、参考にしてほしい。このプログラムでは、近接度のメッセージを表示するように実装している。

アプリケーションを実行すると、【図10】のように近接度に応じたメッセージを受け取れる。

OnEnterイベントは信号エリアに入ったときのみ処理されるが、継続的にBeacon信号を処理したい場合には、OnBeaconProximityイベントを使うとハンドリングが可能である。

3-4. Beaconを使った位置検出「BeaconFance」

次に、Beaconを使った「位置サービス」について考えてみる。前節で大まかな近接度判定について解説したが、位置を検出するにはさらに高度な算出が必要になる。

アプリケーションが動作するデバイスとBeacon機器までの距離を算出する場合、Beaconの出力電波強度（TxPower）と受信時の電波強度（RSSI）を、所定の公式（フリスの伝達公式）で次のように計算する。

＜Beaconとの距離の計算＞
距離＝10^((TxPower − RSSI) / 20)

通常、Beaconの位置情報アプリではこうした計算をプログラムで複雑に組む必要があるが、Delphi/400ではSystem.Beacon.IBEACON.Distanceにメートル単位での距離を自動算出する機能があるので、簡単に距離を割り出せる。

これによってBeacon機器との距離は扱えるが、位置という意味では正確ではない。この距離は、Beacon機器を中心に円形状に発信された距離であり、方向までは特定できない。【図11】

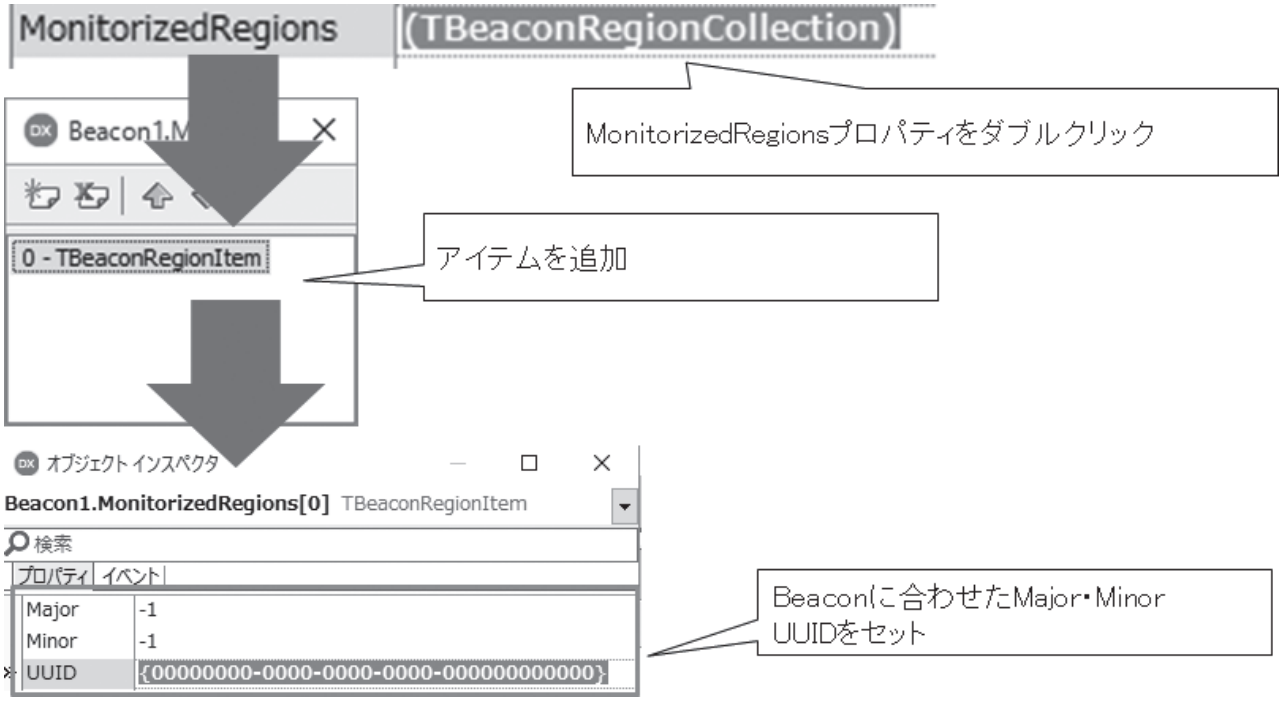
そのため位置測位に関しては、必ず3個以上のBeacon機器からの電波を計算する三角測量技術を使うことになる。

三角測量については算出法が複雑なので、本稿では割愛するが、複数のBeacon機器からの距離を加味した演算を繰り返し行う必要がある。

この位置測位をDelphi/400で簡単に解決するソリューションとして、Delphi開発元のエンバカデロ・テクノロジーズ社が販売する「BeaconFence」というサードパーティ製品がある。コンポーネントとして提供されているが、これを使用すると、こうした位置測位演算ロジックをプログラミングしなくても、ビジュ

図7

図8



ソース1

OnBeaconEnterイベント(Beaconエリアに入った処理)

```
procedure TForm1.Beacon1BeaconEnter(const Sender: TObject;
  const ABeacon: IBeacon; const CurrentBeaconList: TBeaconList);
begin
  Image1.Visible := True; //画像を表示
  ClientDataSet1.Execute; //サーバにデータを登録
end;
```

アルツール上の設定だけで位置サービスのアプリケーションを開発できる。【図 12】

BeaconFence は標準コンポーネントではないが、10 Seattle の GetIt パッケージマネージャに登録されており、簡単にインストールできる。GetIt パッケージマネージャは、［ツール |GetIt パッケージマネージャ］から起動して、対象製品を選択するだけですぐに利用可能である。【図 13】

ただし無償で使用できるのは、対象 Beacon 機器が 3 個までなので、実際の運用時には有償のライセンスが必要である。

4.Beaconの運用

前項でプログラムでの実装ポイントを考察したが、実際の運用ではアプリケーションだけではなく、Beacon 機器自体の知識・利用方法が重要になる。以下では、Beacon 機器の運用上のポイントについて補足する。

4-1. Beacon の運用の注意点

Beacon 機器は主要な規格として iBeacon や AltBeacon があり、Delphi/400 では双方に対応している。

Beacon 機器は信号を発信することを主な機能とするが、プログラムの実装でも触れたように、受信するアプリケーション側は、その信号の強さなどによって距離を判別している。

この信号の強さ（電波）はアプリケーションの動作に大きく影響するので、運用上の考慮点をいくつか把握しておく必要がある。

Beacon 機器の運用で注意すべきポイントは、大きく 3 つある。

1 つ目のポイントは、周波数帯である。Beacon は 2.4GHz の周波数帯を使用する機器であるが、この周波数帯は免許なしで使用できるので、さまざまなデバイスが発している。代表的なものとして電子レンジ、Wi-Fi、一部のデジタルコードレス電話などがある。

Beacon を配置する場所では、こうした機器が使用しているチャネルが重複しないよう配慮する必要がある。どういった周波数が使われているかは施設側に確認するか、あるいは専用アプリケーションなどで測定もできる。テストしたうえで、競合していないことの確認が必要である。

2 つ目のポイントは、モノによる電波の干渉である。たとえば電波が壁にぶつかれば、反射した電波も存在するので、場所によっては強い電波や弱い電波が混在する。また水を多く含む物体（生物を含む）は、電波を吸収しやすい。たとえば Beacon とデバイスの間に人がいると、電波が吸収され、正しく電波を受信できない可能性もある。【図 14】

そのため施設によっては、人が電波の直線上に入らないよう天井などの上に配置することも多い。ただしその場合は、天井からの距離も電波に影響することを考慮する必要がある。

3 つ目のポイントは、電力の確保である。Beacon 機器の電力供給方法には、主に電池型と給電型がある。【図 15】

通常、Beacon の設置は電源が近ければ給電型が便利だが、位置情報を必要とする場所では、電源が確保できない場合も多い（たとえば屋外のイベントなど）。そうしたケースでは、電池型がよく採用されている。

電池は製品にもよるが、大体 1 年ぐらいはもつ。数が多いと残量チェックなどは現実的ではないので、余裕のある定期交換を前提に運用を考えるのが一般的である。

4-2. Beacon のセキュリティ

Beacon のセキュリティ面について、考えてみる。

前述したように、Beacon は UUID、Major、Minor などの情報で識別されるが、その識別情報を知っていれば、意図的に複製できる。

iPhone では、iPhone 自体を Beacon 機器として利用するアプリケーションが AppStore で配布されている。プログラム開発では、そうしたアプリケーションで Beacon のテストが疑似的に行えるメリットがある。しかしこれは同時に、実際の Beacon 機器についても「なりすまし」が可能なことを意味する。

たとえばクーポンの Beacon 信号を複製されると、不正にそのクーポンサービスを利用される危険がある。【図 16】

社内や工場内のシステムなどでは、あまり考慮する必要はないかもしれないが、Beacon で公共に信号を発信する場合には、機密性の高いものは扱わないように注意すべきである。

もしセキュリティが必要な場合は、単純に Beacon 信号を受発信するだけでなく、アプリケーションが動作する条件をシステム的にガードする必要がある。たとえば並行してユーザー情報をチェックしたり、ログ出力で重複利用を防ぐなどの方法もセキュリティ的に有効である。

5.まとめ

本稿では、Beacon 機器の特徴や基本情報の確認、IoT での活用例などを考察してきた。また、そうした Beacon 機器をシステム活用するためのプログラム実装も、Delphi/400 では専用コンポーネントにより非常に簡単であることを説明した（通常は Bluetooth 情報を受け取る部分から、すべて独自のプログラムとして開発する必要がある）。

Beacon も今後はさらに機能が進化し、機器自体も低コスト化していくと思われる。たとえば運用上の考慮点として、電力について説明したが、現在では太陽電池などで、交換電源を必要としない Beacon 機器も開発されている。

この Beacon 技術は、東京駅構内でのナビゲーションサービスの実証実験でも使われており、これからの IoT 技術として大きな期待が寄せられている。【図 17】

Beacon はユーザー操作を必要としないので、浸透すれば多くの施設やシステムで利用できる可能性を秘めている。まだ技術展開が始まったばかりの IoT 分野だが、今までにないユーザーインターフェースやビジネスモデルを考えていくうえで、Beacon は非常に画期的で興味深い。Delphi/400 では簡単に実装できるので、IoT に取り組む第一歩として、システム開発に活用していきたい。

M

ソース2

OnBeaconExitイベント(Beaconエリアから出た処理)

```
procedure TForm1.Beacon1BeaconExit(const Sender: TObject;
  const ABeacon: IBeacon; const CurrentBeaconList: TBeaconList);
begin
  Image1.Visible := False; //画像を非表示
end;
```

ソース3

OnCreateイベント(初期処理)

```
procedure TForm1.FormCreate(Sender: TObject);
Begin
  Image1.Visible := False; //画像を非表示
  Beacon1.Enabled := True; //Beaconを有効化
end;
```

図9

ソース4

OnBeaconEnterイベント(Beaconエリアに入った処理) 応用

```
procedure TForm1.Beacon1BeaconEnter(const Sender: TObject;
  const ABeacon: IBeacon; const CurrentBeaconList: TBeaconList);
begin
  Image1.Visible := True;
  ClientDataSet1.Execute;
  //近接度を判定
  case ABeacon.Proximity of
    //0,5m未満
    TBeaconProximity.Immediate:
    begin
      ShowMessage('近接');
    end;
    //0.5m以上、1.5m以下
    TBeaconProximity.Near:
    begin
      ShowMessage('近い');
    end;
    //1.5mより遠い
    TBeaconProximity.Far:
    begin
      ShowMessage('遠い');
    end;
    //距離判定不能
    TBeaconProximity.Away:
    begin
      ShowMessage('測定不能');
    end;
  end;
end;
```

図10

図11

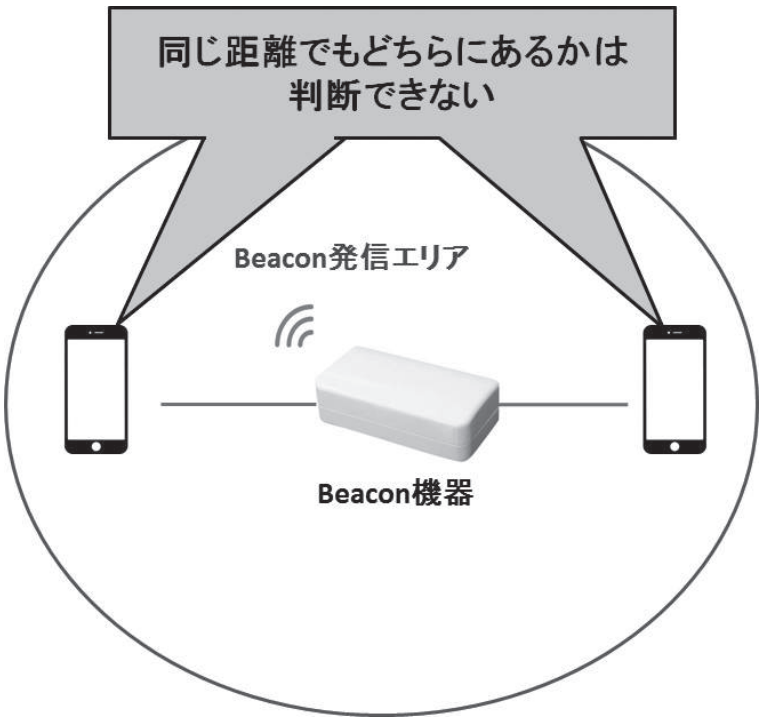
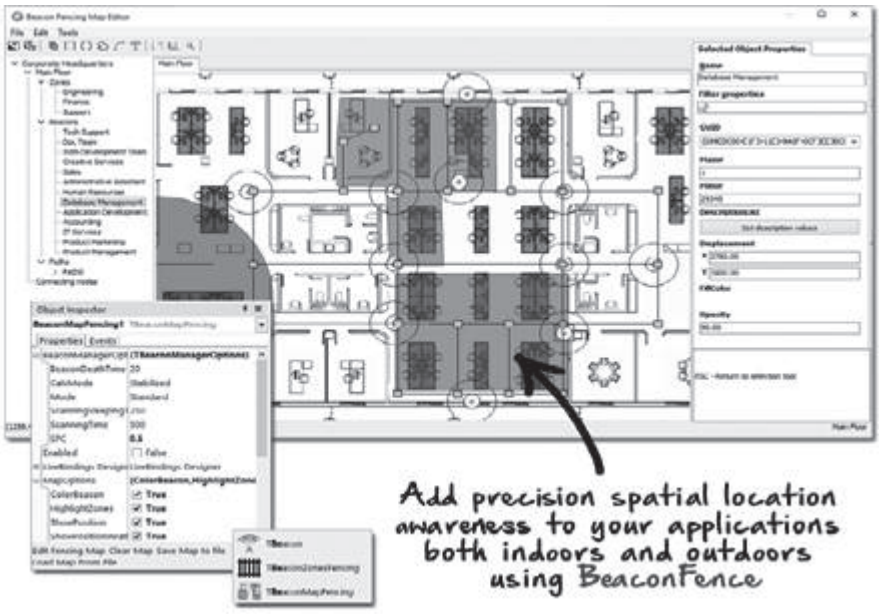


図12



出典：Embarcadero Technologies BeaconFence

図13


GetIt パッケージ マネージャ
検索
絞り込み
すべて
購入可能
無料
購入済み
インストール済み
ソート順
カテゴリ
All
Abbrevia 10.0
TurboPack
インストール
AsyncPro for VCL 1.0
TurboPack
インストール
BeaconFence 1.0
Embarcadero Technologies
今すぐ試す
インストール
完了!
閉じる
選択するだけで
自動インストールや
自動アンインストールが可能

図14


配置によって人やモノが電波を弱めてしまう

図15


電池型


給電型

図16

図17

出典：ジャパンスマートナビ：国土交通省