ホームネットワークとOSGi

2008MI011 朝倉知也
2008MI079 岩井　大

目次

1.1 OSGi – 背景 –

ホームネットワークでの利用を前提に誕生

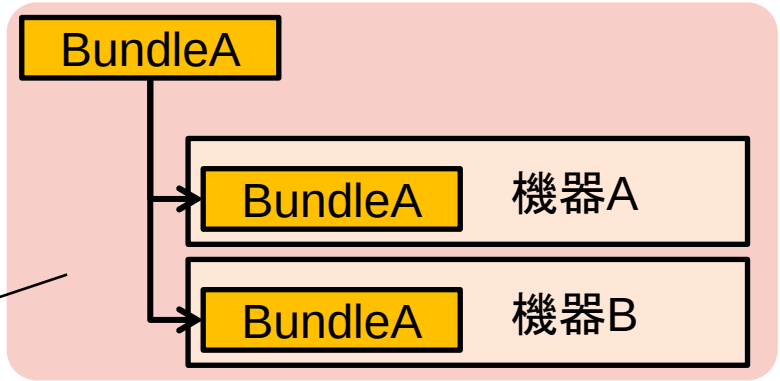
汎用性の高さから，現在では多分野で活用

自動車，携帯電話等

BundleはJava言語で実装(プラットフォーム非依存)

・他の組込み機器でも修正せずに動作可能
・他のソフトウェア開発の際に再利用可能

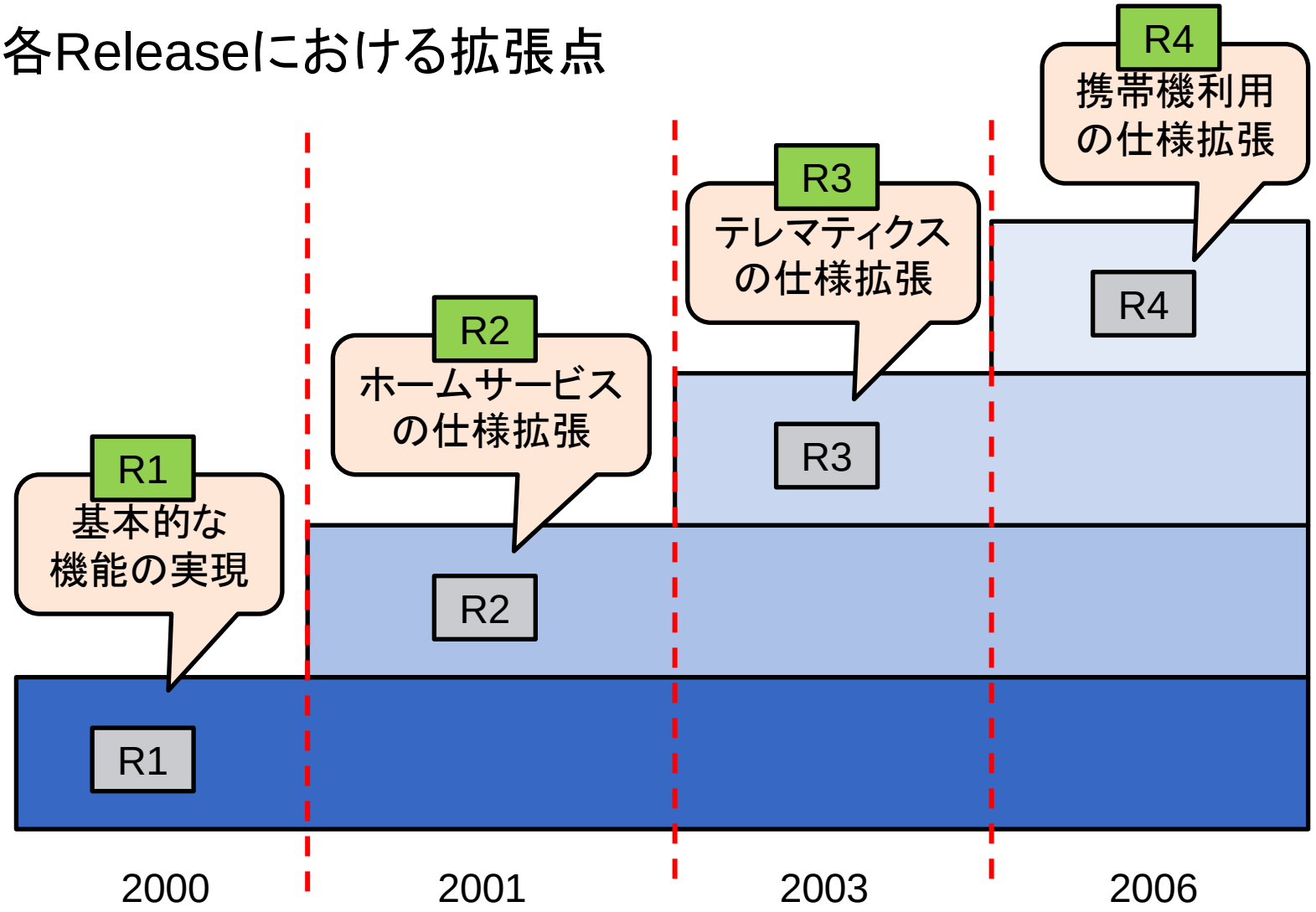
開発の効率化



BundleAをそのまま
他の機器で利用可能

1.2 OSGi – 現状:Release(1/2) –

各Releaseにおける拡張点



1.2 OSGi – 現状:Release(2/2) –

- Release3 - テレマティクス(自動車)への対応

  近年，自動車に搭載されるソフトウェア量は増加し，搭載される資源も増加傾向にある

  → しかし，会社や車種ごとに環境・プラットフォームが異なる為，効率よく開発可能な共通プラットフォームが必要であった

  → BMW社が中心となり，自動車向けの共通Bundleを制定し，Release3と同時に公開

- Release4 - 携帯電話への対応

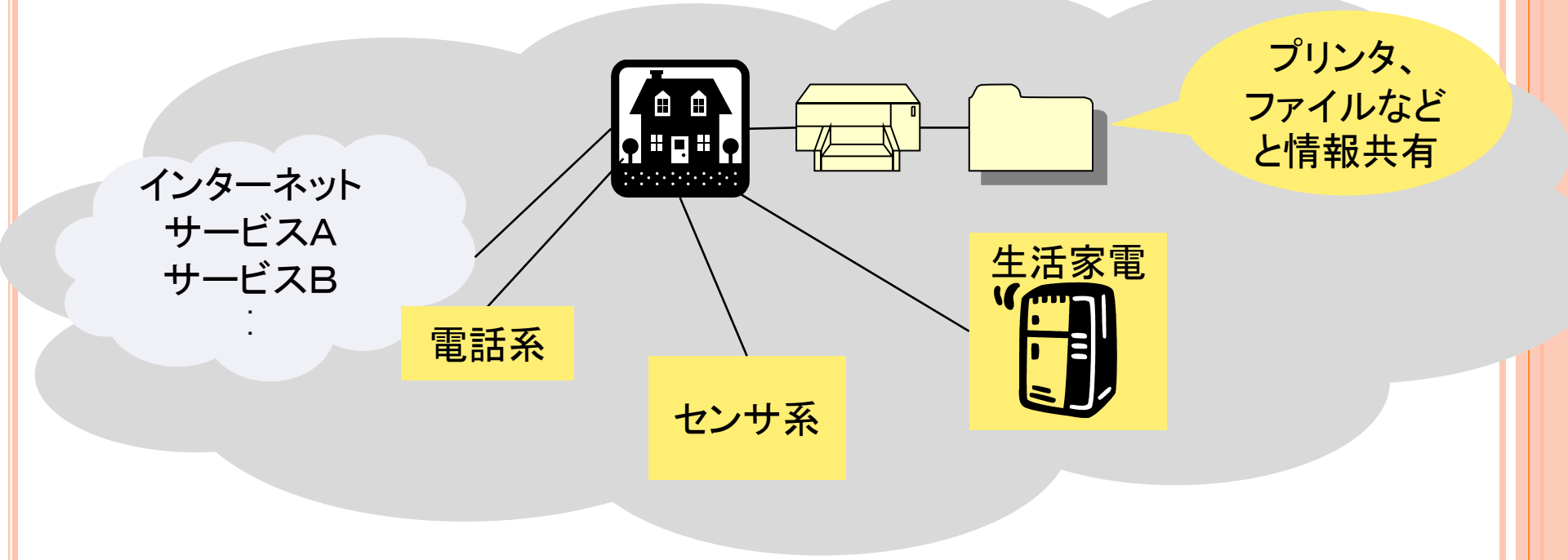
  携帯電話では以前よりJavaが利用されていたが，アプリケーションを単一ファイルとして実行していた

  → OSGiを利用することで，アプリケーション(Bundle)間での連携を可能とし，開発・管理をより効率的にした

2.1 ホームネットワークとは(1/2)

- ホームネットワークとは・・

  家庭内に構築したPCなどの通信システムのLAN環境のこと

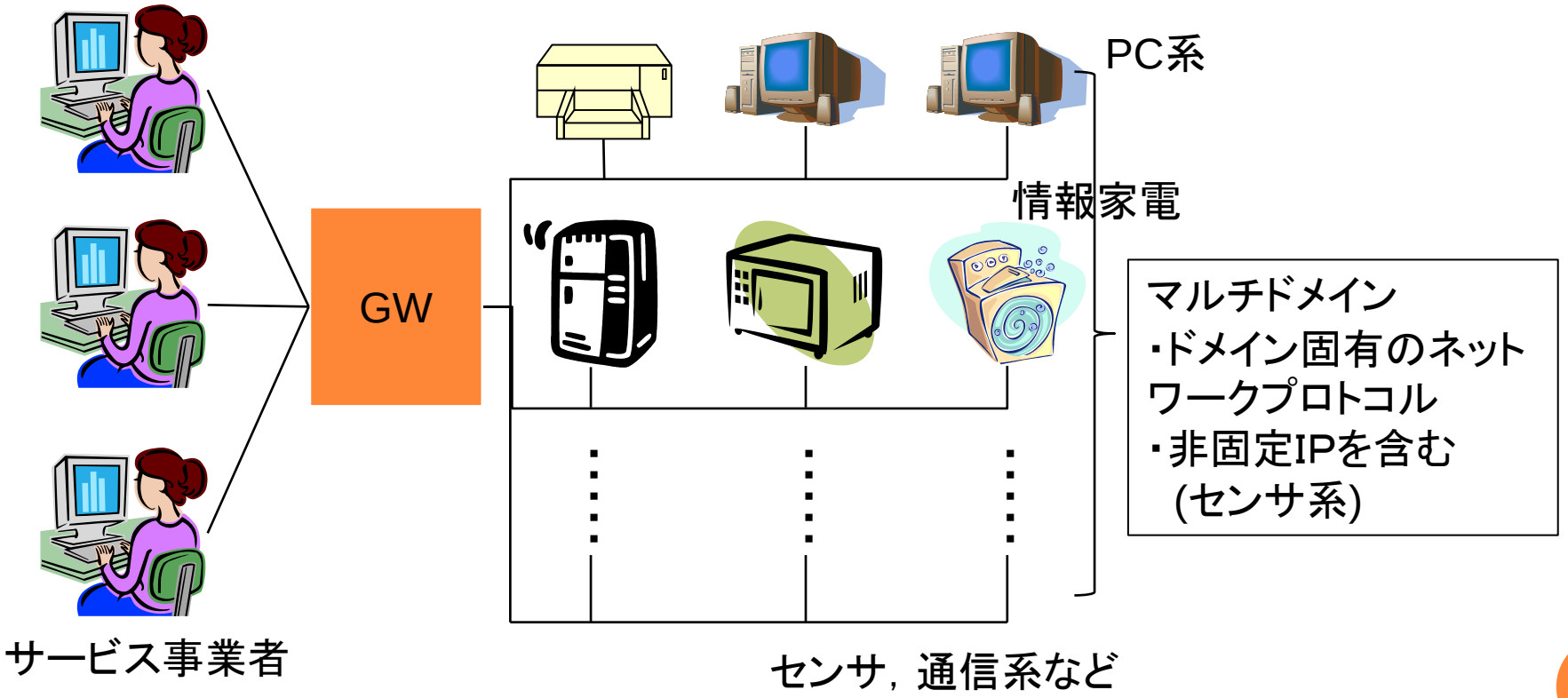


時代が進むにつれて，サービス事業者，デバイス/機器，ネットワーク技術が多様化してきた

→各サービス事業者が個々に独自のゲートウェイを使いサービスを行っている

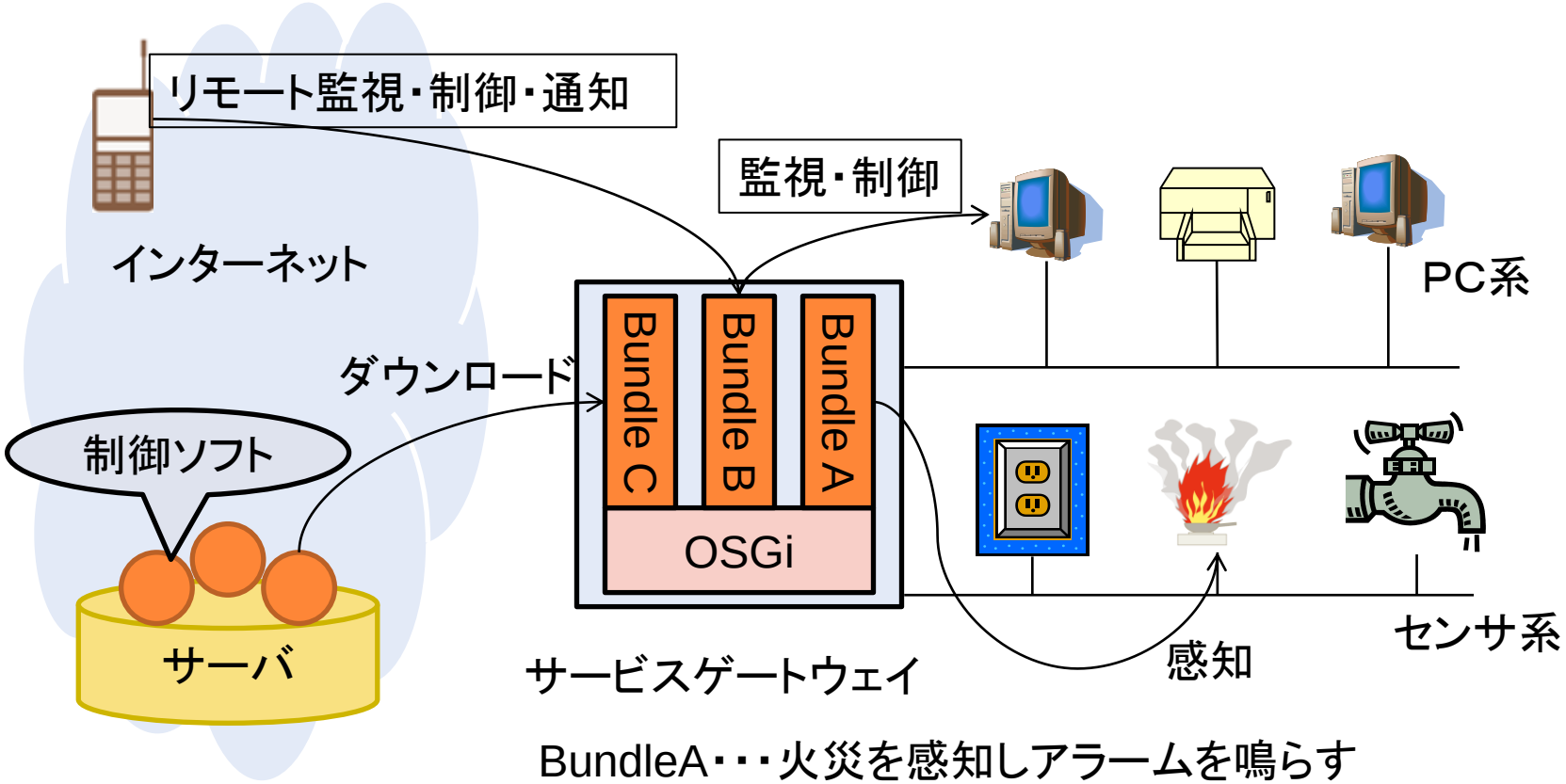
2.1 ホームネットワークとは(2/2)

- 各サービス事業者が個々にGW機器を設ける
  →一つのGWによってサービス事業者と機器を繋ぐ
  「マルチドメイン＋ゲートウェイモデル」



2.2 ホームネットワークにおけるOSGi

例)家で火事が起きたとき，アラームや外部に通知する
　システムのOSGiを使った制御モデル



BundleA・・・火災を感知しアラームを鳴らす
BundleB・・・センサの監視や外部に情報を送る

3.1 OSGi – 実装・サービス連携(1/7) –

- 実装するサービス

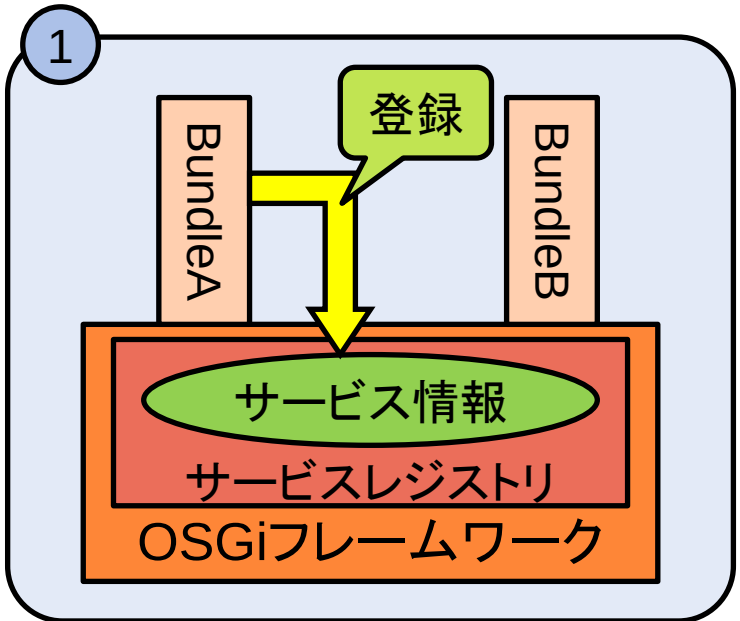
  与えられた文字列に”_test” を追加して出力する

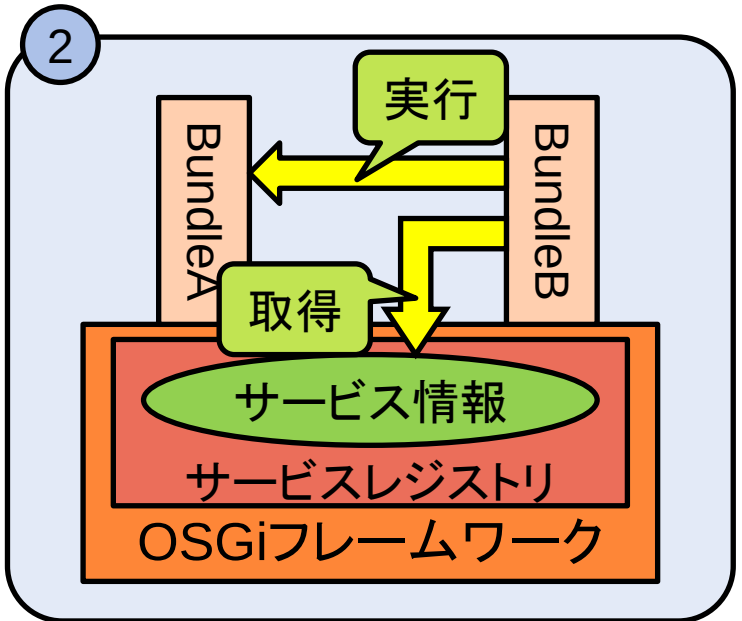
  ex) 引数に”ABC” → 出力:”ABC_test”

- 実装・実行手順

  1.上記のサービスをBundleAでサービスとして実装し，登録する

  2.BundleBにて登録されたサービスを取得，実行する



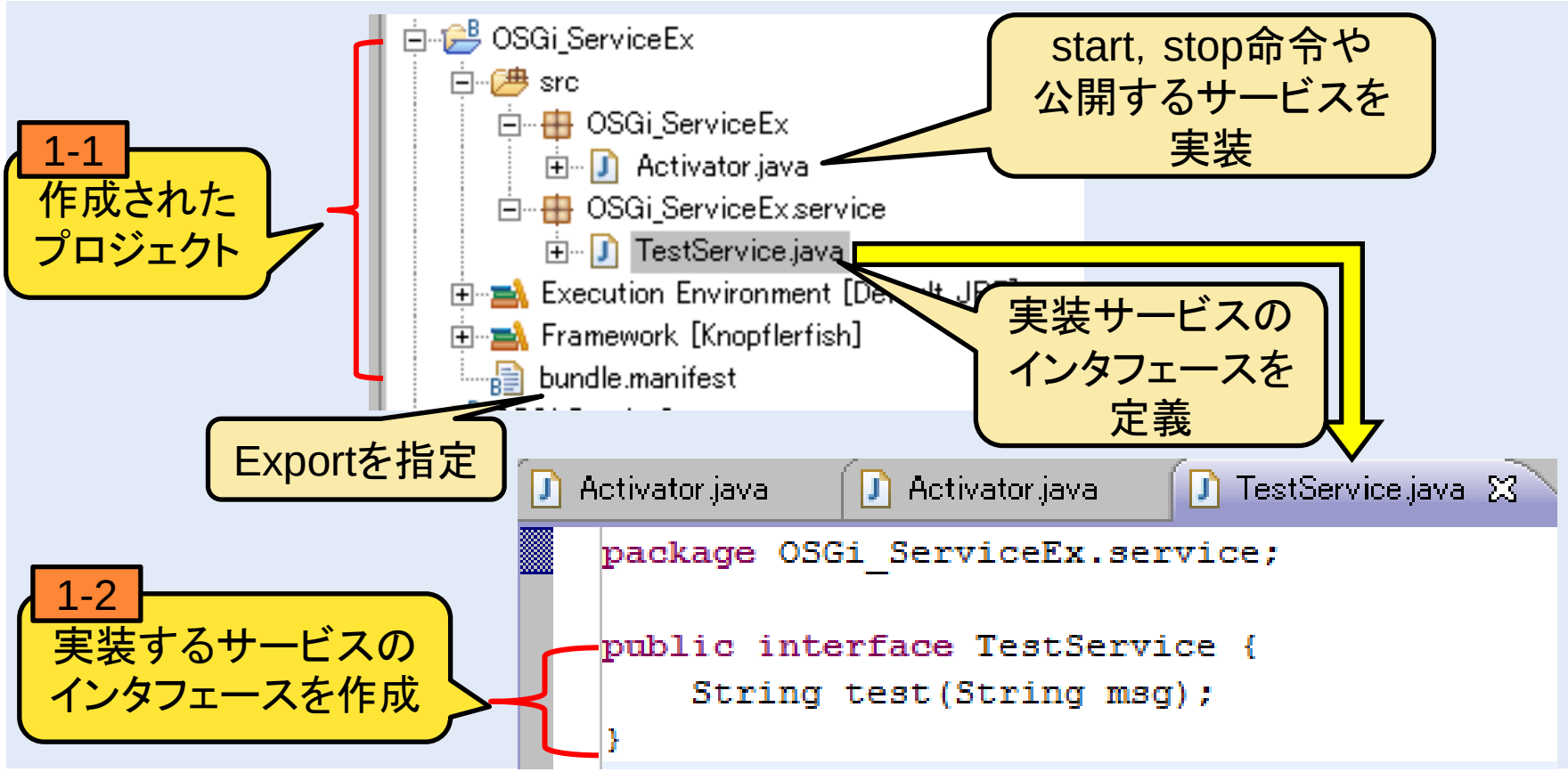


3.1 OSGi – 実装・サービス連携(2/7) –

1.BundleAの作成

1-1.前回の実装と同様にBundleプロジェクトを作成

1-2.実装するサービスのインタフェースを作成する



3.1 OSGi – 実装・サービス連携(3/7) –

1-3.Activator.java – Activatorクラス内でサービスを実装する

1-4.Activator.java – Activatorクラス内のstartメソッドで
実装したサービスを登録するプログラムを作成

1-4
サービス登録

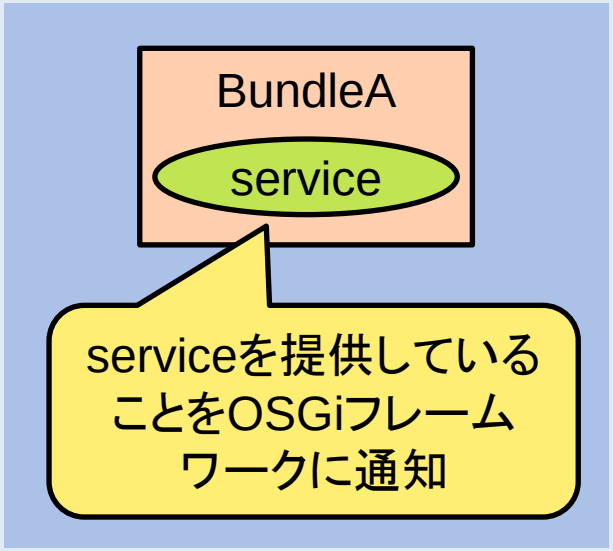
1-3
サービスの
実装

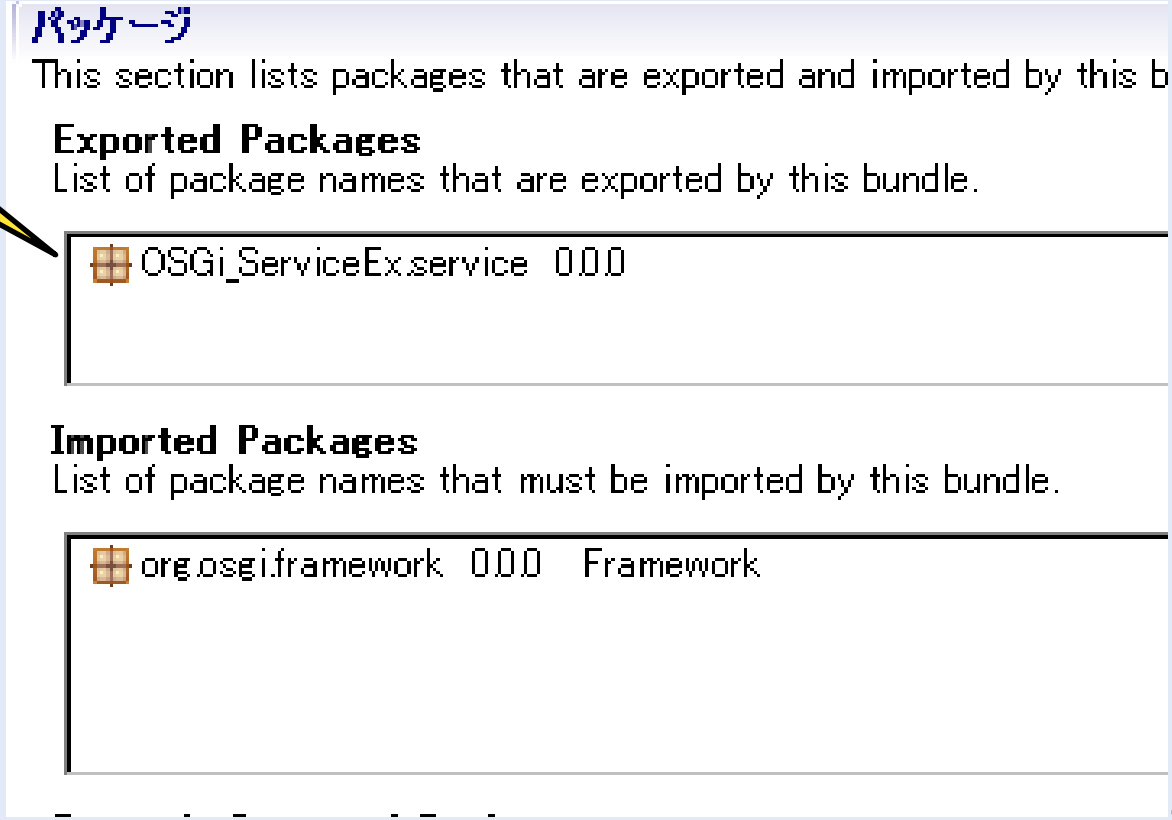
```
public class Activator implements BundleActivator {

  /* (non-Javadoc)
   * @see org.osgi.framework.BundleActivator#start(org.osgi.framework.BundleContext)
   */
  public void start(BundleContext context) throws Exception {
      context.registerService(TestService.class.getName(), new TestServiceImpl(), null);
  }

  /* (non-Javadoc)
   * @see org.osgi.framework.BundleActivator#stop(org.osgi.framework.BundleContext)
   */
  public void stop(BundleContext context) throws Exception {
  }

  private class TestServiceImpl implements TestService{
      public String test(String msg){
          return msg + "_test";
      }
  }
}
```

3.1 OSGi – 実装・サービス連携(4/7) –

1-5.実装したサービスのインタフェースをマニフェストファイルの
Exported-Packagesに指定する
(Bundle間の依存関係を明確にする為)

1-5
サービスのインタフェースを指定





3.1 OSGi – 実装・サービス連携(5/7) –

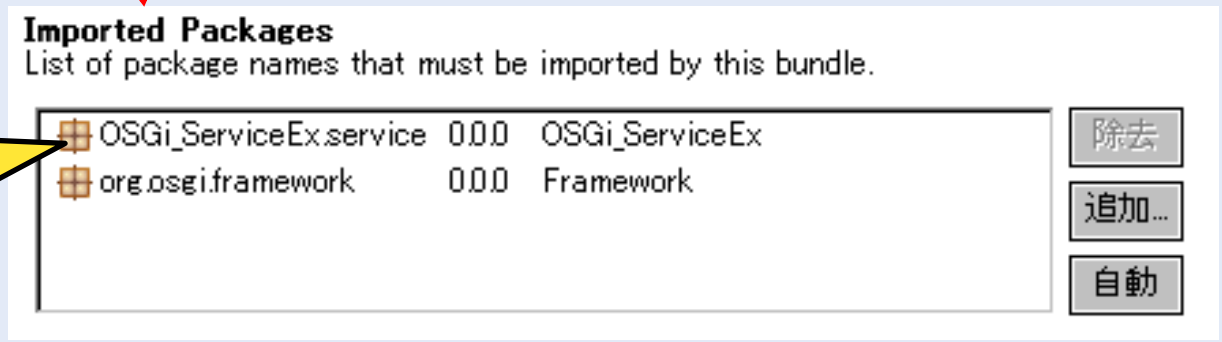
2.BundleBの作成

2-1.BundleAと同様にプロジェクトを作成

2-2.マニフェストファイルでImportを指定する

2-1
作成された
プロジェクト

2-2
先程のインタフェースを
インポート



3.1 OSGi – 実装・サービス連携(6/7) –

2-3.Activator.java – Activatorクラス内のstartメソッドにて
サービスを検索，取得し実行するプログラムを作成

```
public void start(BundleContext context) throws Exception {
    ServiceReference sr = context.getServiceReference(TestService.class.getName());

    if(sr != null){
        TestService service = (TestService)context.getService(sr);
        if(service != null){
            System.out.println("test service result : " + service.test("ABC"));
            context.ungetService(sr);
        }
    }
}
```

引数で指定されたインタフェースを実装するサービスリファレンスを取得

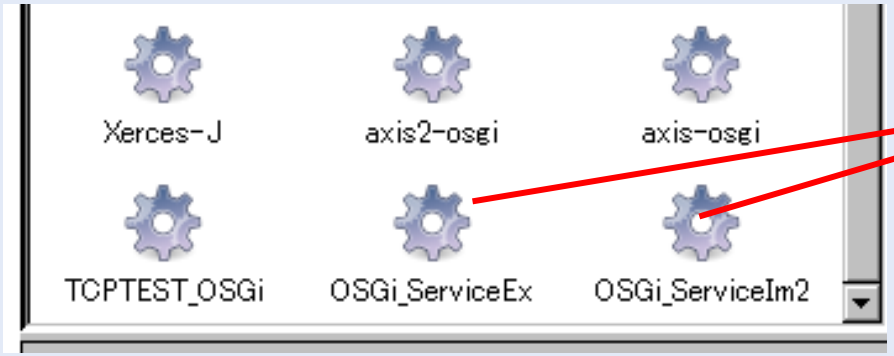
サービスリファレンスに該当するオブジェクトを取得

serviceのtestメソッドを実行(引数:”ABC”)

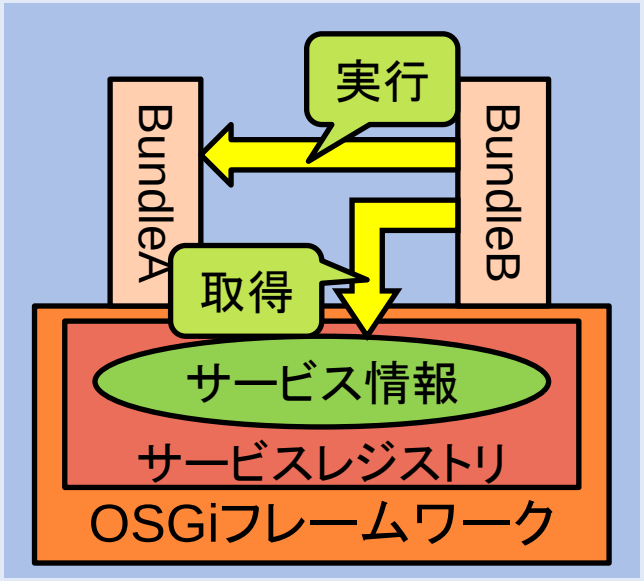
3.1 OSGi – 実装・サービス連携(7/7) –

3.エクスポート・実行

作成したBundleA,Bをエクスポートし，Knopflerfishにインストールして実行する(BundleAを実行してからBundleBを実行)



エクスポートされた
BundleA(OSGi_ServiceEx),
BundleB(OSGi_ServiceIm2)



BundleAをstartしてから
BundleBをstartし，仕様通りに
実行されたことを確認

```
[stdout] test service result : ABC_test
```

4 今後の課題

- 今回はOSGiの背景から調べたが，次回からは再び自動車におけるOSGiについて深く調べたい.
- OSGiの簡単な実装としてHelloWorld とサービス連携について実装したので，次は基本Bundleを用いた実装を行ってみる.

5 参考文献

- 情報家電ネットワークと通信放送連携

  IPTVで実現する家庭内ユビキタス　阪田史郎編著
- 濱千代正弥、片桐雅仁：自動車ネットワークサービスの連携アーキテクチャ