ハードウェア実験

組み込みシステム入門
第9回

2010年11月18日

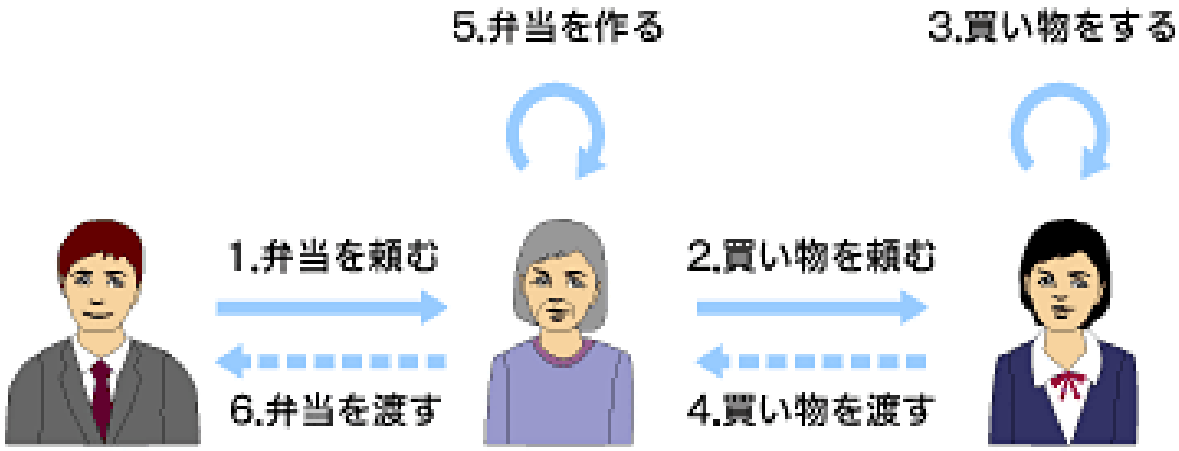
状態遷移の設計と記述

前回に引き続き

- ICトレーナ上のスイッチを読み込みます。
- 複数のスレッド間で、データの受け渡しを行うプログラムを課題とします。
  - 4番目のサンプルプログラム(Buttonプロジェクト)を改変します。
- 「状態機械」の動作をプログラムしてみます。

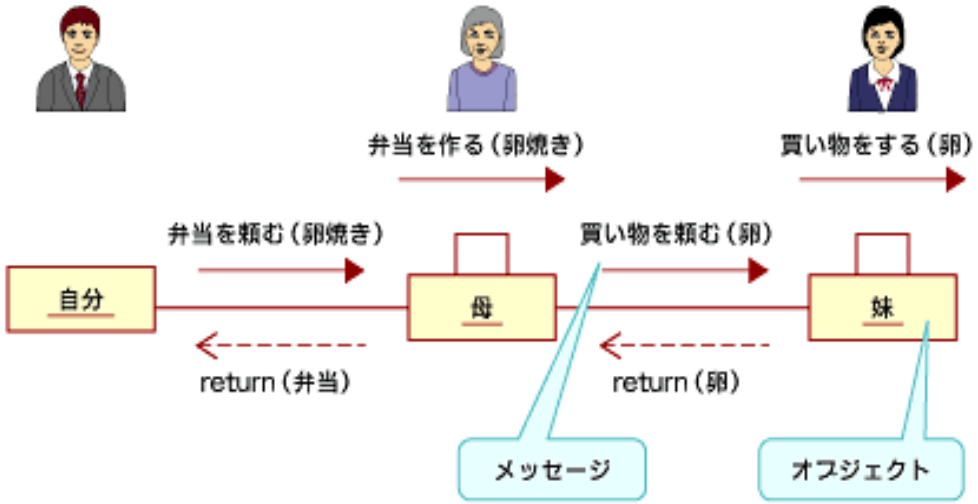
相互作用図

- 自律分散協調動作と相互作用図
- http://www.atmarkit.co.jp/im/carc/serial/object06/object06.html
  - オブジェクトは1つ1つは自律
  - それらが「メッセージを通して、協調しながら全体として1つの仕事をする」というのがオブジェクト指向の自律分散協調動作の考え方です。
- シーケンス図とコラボレーション図とがある。



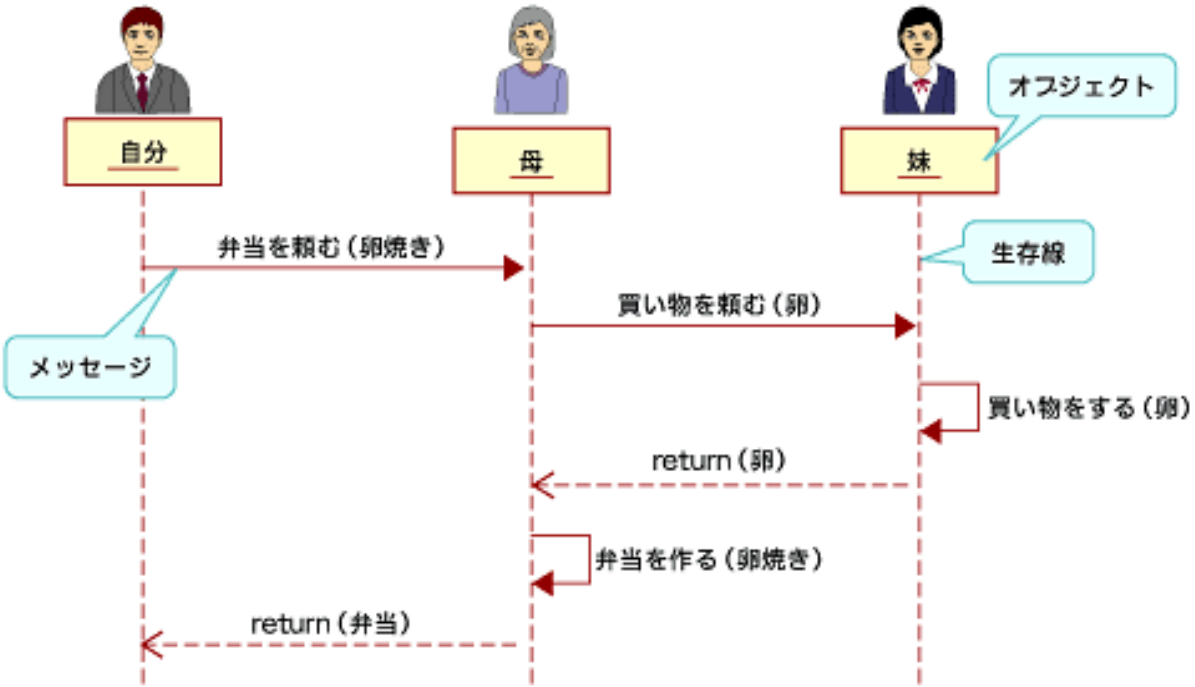
コラボレーション図

- 「自分」から「母」に「弁当を頼む（卵焼き）」というメッセージを送る
- 「母」から「妹」に「買い物を頼む（卵）」というメッセージを送る
- 「妹」は自分自身に「買い物をする（卵）」というメッセージを送る（メソッド）
- 「妹」はメッセージ2の依頼により、買ってきた卵を「母」に渡す
- 「母」はメッセージ1の依頼を達成するため、自身に「弁当を作る（卵焼き）」というメッセージを送り、完成したら弁当を「自分」に渡す



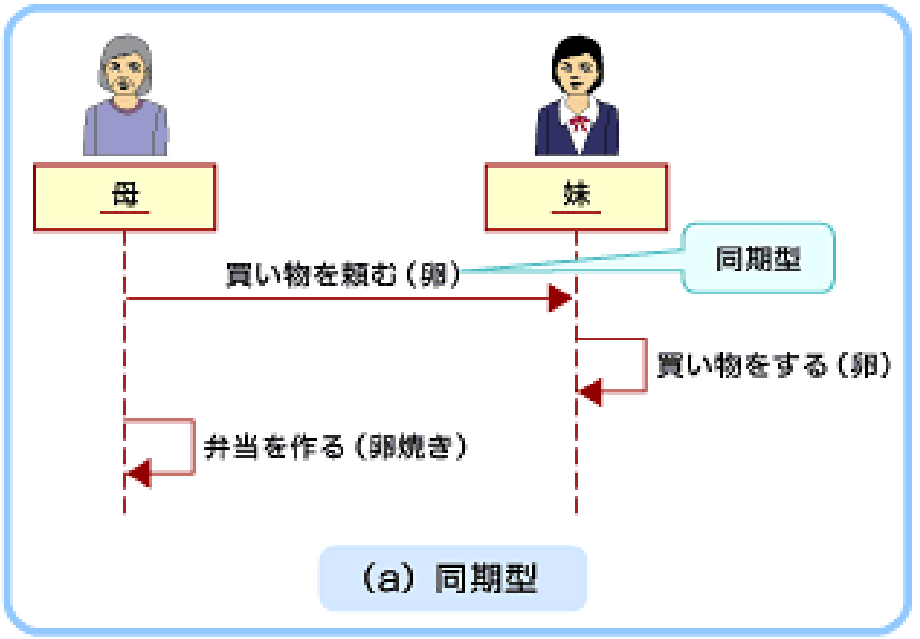
シーケンス図

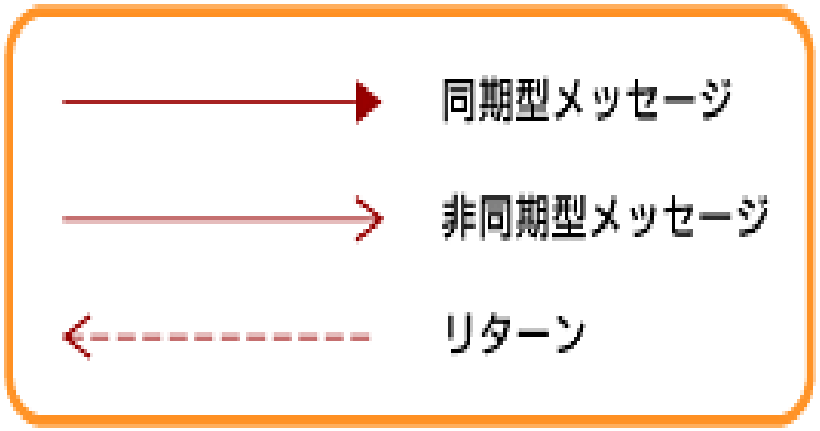
- メッセージの順番を時系列で表現する
- オブジェクトを上に並べ、その下に生存線と呼ばれる点線を引く
- メッセージはこの生存線の間に、シナリオの時間の流れに沿って上から順に並べる

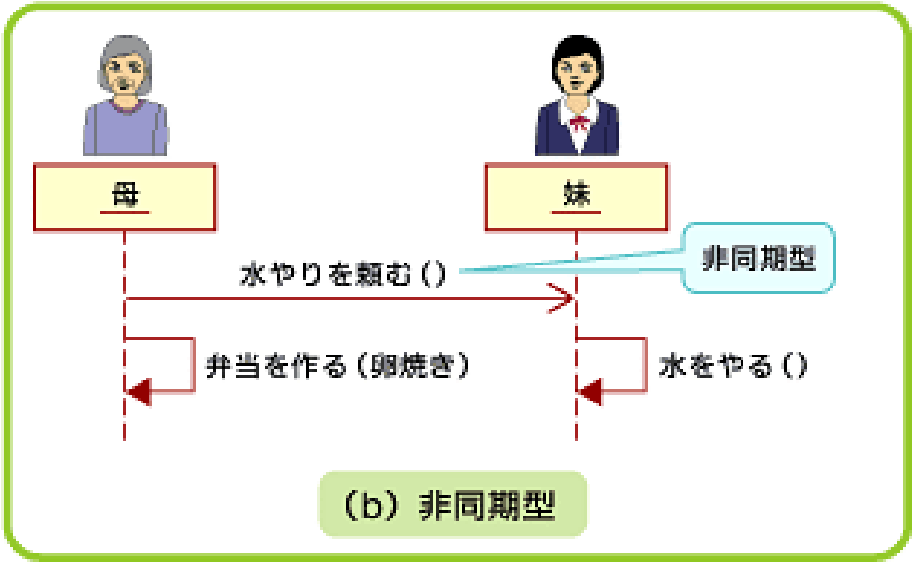


メッセージの書き方

- 同期型と非同期型がある。
- リターンを待つ場合と待たない場合

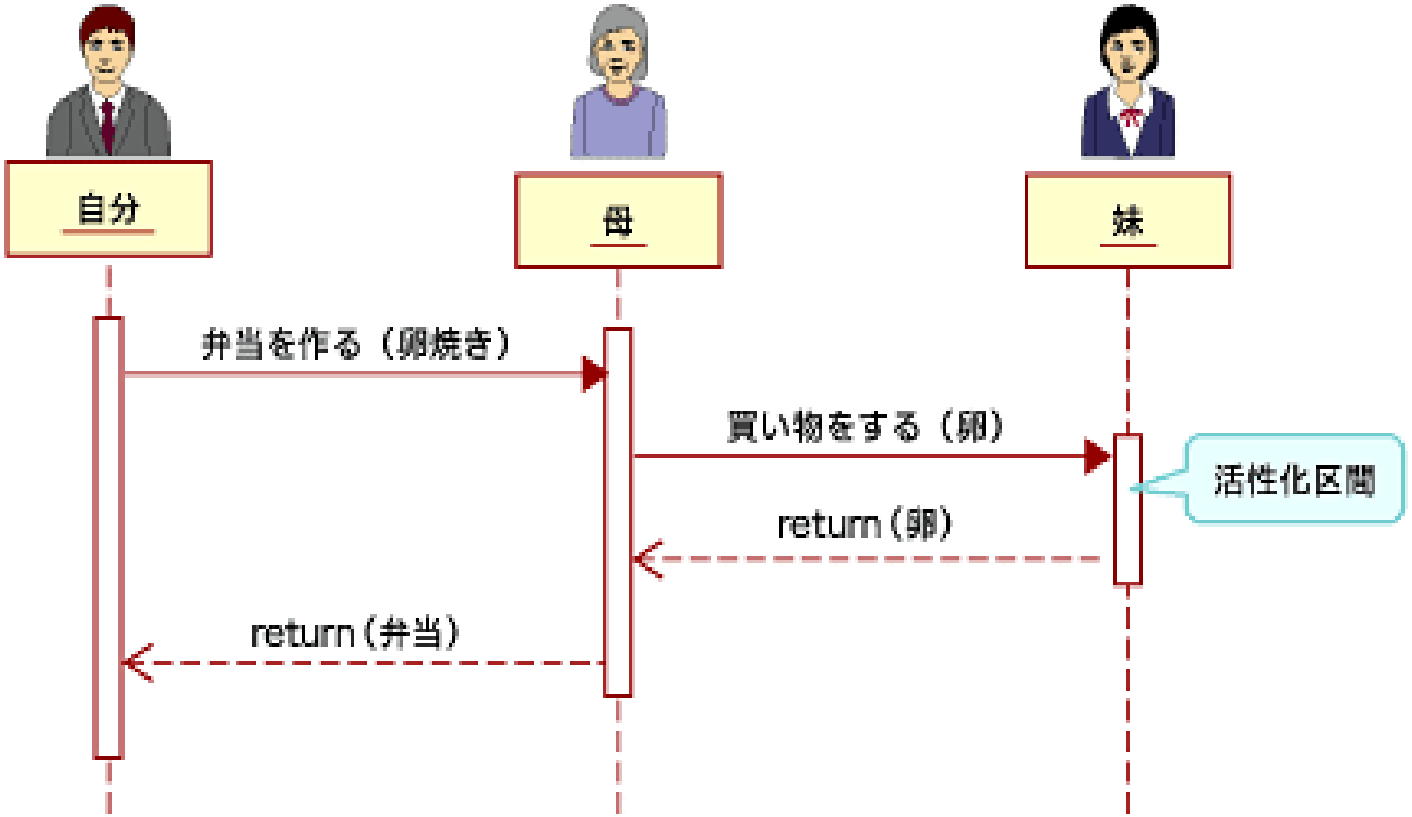






活性化区間

- スレッドが「起きて」いる期間を明記する。
- 生存線上の長方形は、呼び出された操作がアクティブな時間を表すもので、活性化区間または制御フォーカスと呼ぶ。



button projectでのEvent

- caseで記述されている行
  - 43行目
    - タイマー・イベントを受け付けている。
    - case t when timerafter(time) :> void:
  - 56行目
    - 通信チャンネルからのLEDの値
    - case c :> ledVal :
  - 71行目
    - ボタンが押されたイベント
    - case button when pinseq(0) :> void:

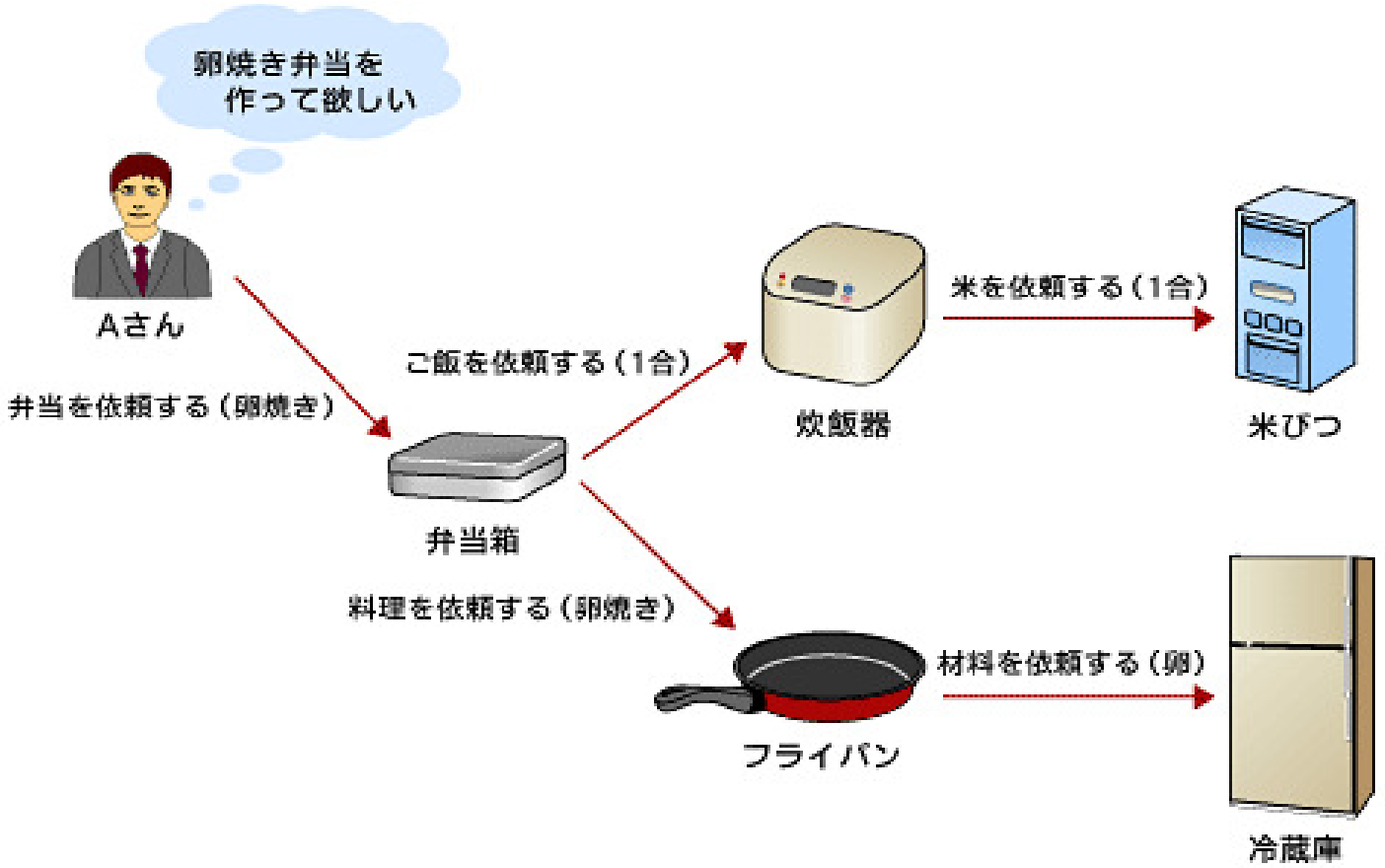
ステートチャート図のシナリオ

- http://www.atmarkit.co.jp/im/carc/serial/object07/object07.html
- ここから始めるオブジェクト指向
- Aさんは卵焼き弁当を欲しいと思ったとします。モデルに登場するオブジェクトは弁当箱、炊飯器、米びつ、フライパンおよび冷蔵庫です。冷蔵庫には卵が入っています。これらのオブジェクトが協力してAさんの要求をどのように達成すればよいかを、オブジェクトを擬人化してちょっと遊び心で考えてください。

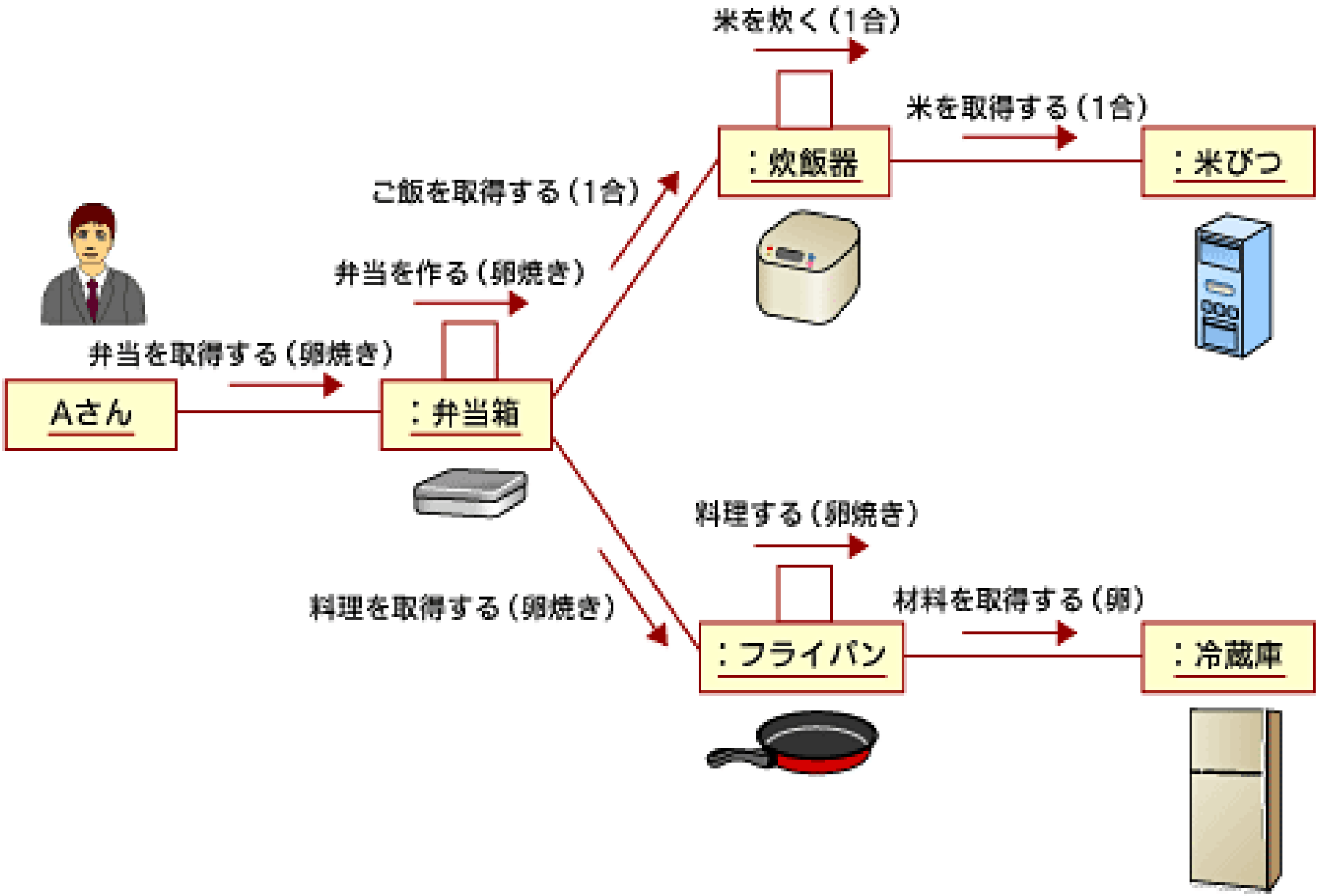
シナリオの図式化

- Aさんは弁当箱さんに「卵焼き弁当を作ってください」とお願いします。弁当箱さんは炊飯器さんのところに行ってご飯を1合お願いします。

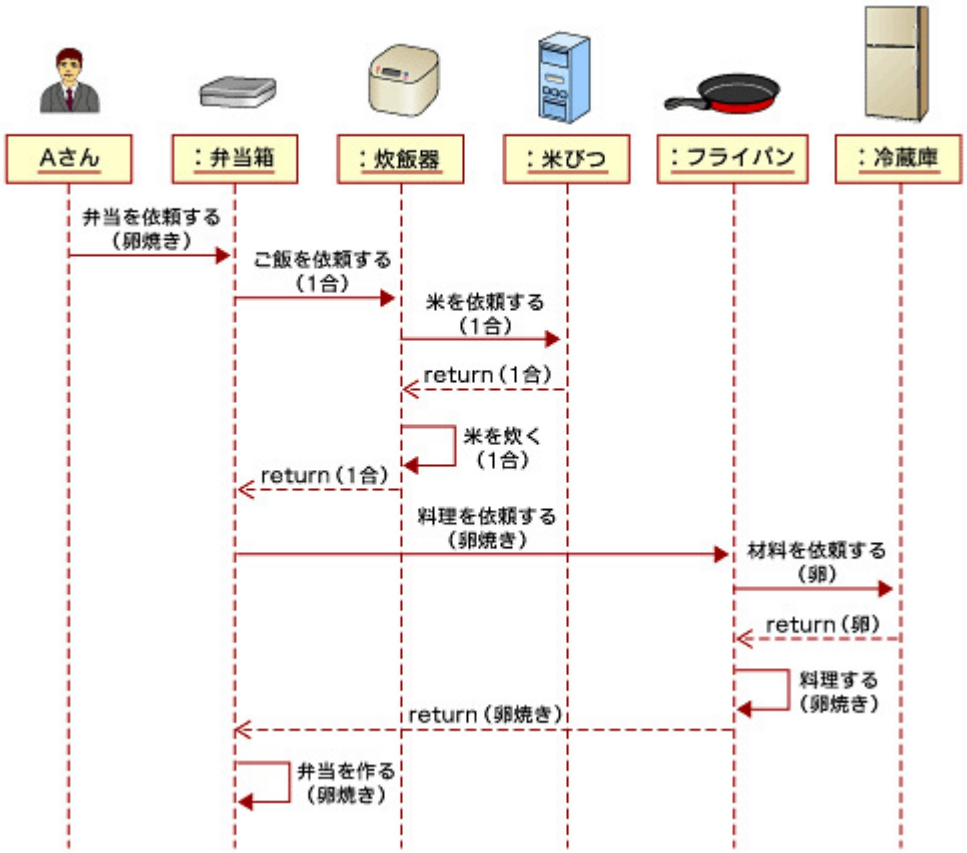
（以下略）



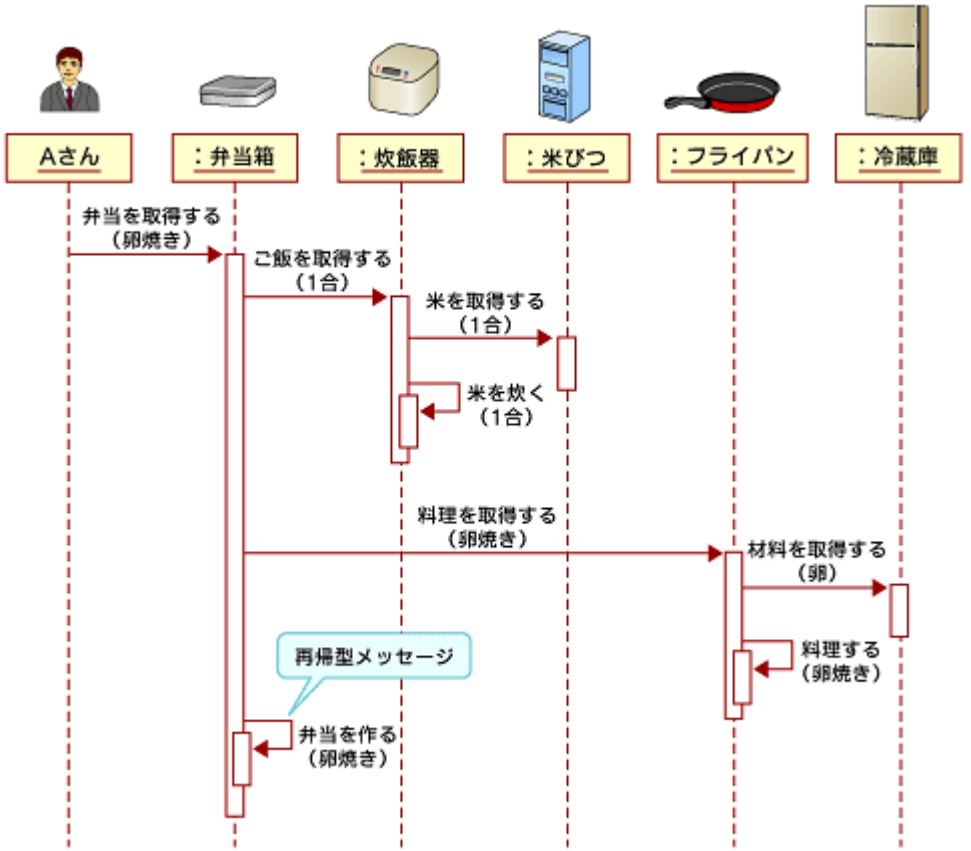
シナリオのコラボレーション図



シナリオのシーケンス図

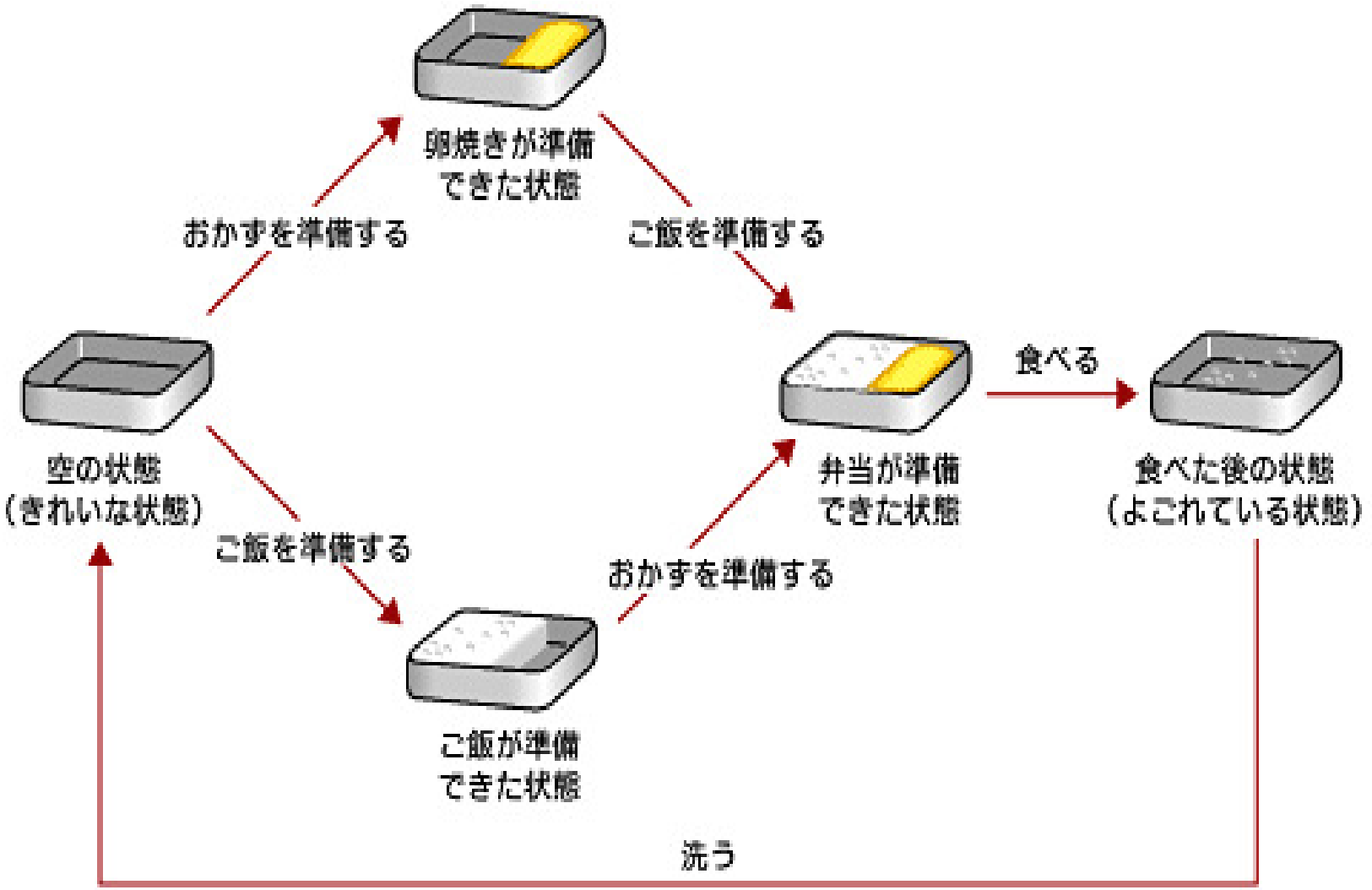


リターンを書き入れた例

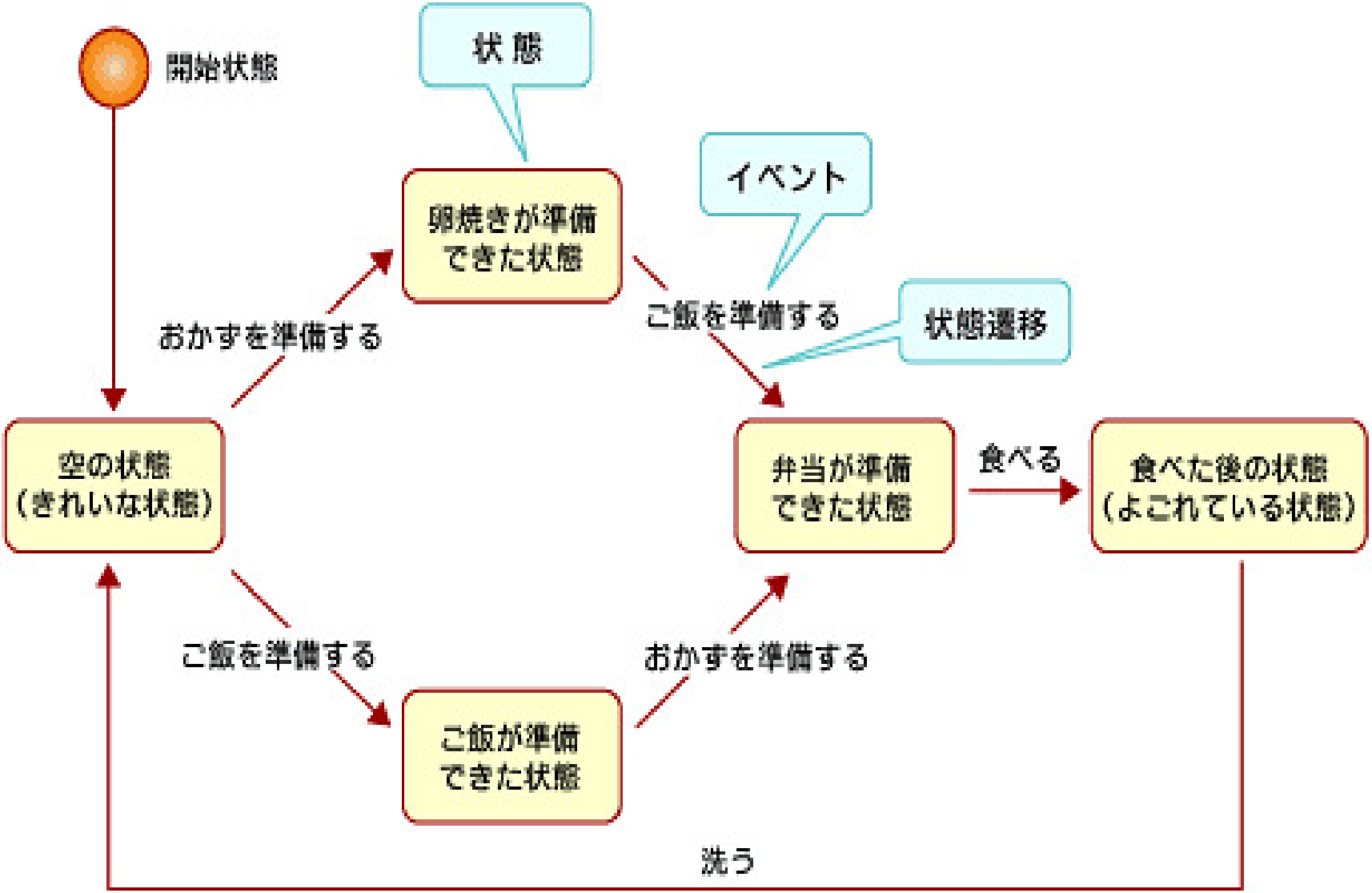


活性化区間を書き入れた例

弁当箱の状態変化


卵焼きが準備
できた状態
おかずを準備する
ご飯を準備する
空の状態
（きれいな状態）
食べる
弁当が準備
できた状態
食べた後の状態
（よごれている状態）
ご飯を準備する
おかずを準備する
ご飯が準備
できた状態
洗う

弁当箱のステートチャート図



前回のスライド(復習)

- 何がEventか？
- なぜ、buttonListenerだけ別のスレッドにしたか？

何がEventか？

- システム側から見て、予期できない事象
  - ボタンが押された
    ⇒　人が操作している
  - 通信チャンネルからの値の受け取り
    ⇒　他のスレッドの動作は、関知しない
  - タイマーが予定時刻になった
    ⇒　時計をじっと見ている訳ではない。
- ※ タイマーは「わかる」のではないか？
  - じっと見ていればわかるが、それではタイマーの意味がない。（他の作業をしたいから、アラームや目覚まし時計を設定する！）
  - タイマーを一度セットしたら、タイマーからの通知は受け付けるものの、他の作業をするので「じっと監視」はしない！

なぜ、buttonListenerだけ別のスレッドにした？

- buttonListenerの①のSelectでは、ボタンが押されるまでsleepする。
- このcase文は、他のEventと同じselectに書くことができる。
- ところが、一度ボタンが押されると②の部分でsleepし、ボタンが離されるまで待つ。
  - ここで待ってしまうと、Selectの他の選択肢が処理できない。

```
64 void buttonListener(in port button, chanend c)
65 {
66   int led = 1;
67   while (1)
68   {
69     select                                    ①
70     {
71       case button when pinseq(0) :> void:
72       {
73         c <: led;
74         led = (led + 1) & 0xF;
75
76         button when pinseq(1) :> void;        ②
77
78         break;
79       }
80     }
81   }
82 }
83
```

このために、button処理のpinseq は、タイマーなどからスレッドを分けて、独立したスレッドで走らせている。

前回出題した報告課題

- 以下のプログラムを書いて下さい。
- 「点滅中」と、「待機中」の二つの状態を作る。
  - 「待機中」は、XK-1上の4つのうち特定のLEDが点灯。
  - 「点滅中」では、さらに「点灯中」と「消灯中」の二つの状態があり、交互に点灯と消灯を繰り返す。
- ICトレーナ上のスイッチ（任意の一つ）がONになったら、「点滅中」になり、OFFになったら「待機中」になるようにする。
  - ICトレーナ上の一つのスイッチで、LEDを切り替える。
- 発展課題A
  - XK-1上のスイッチ1を押すとLEDがカウントアップするようにする。
  - 1回押すごとに、LED表示の4ビット値がカウントアップする。
- 発展課題B
  - XK-1上のスイッチ2を押すと、点滅の周期が変化するようにする。
  - 点滅の周期が一定以上短くなったら、最初の状態に戻す。

課題の内容をシナリオ化する

- ボタン1、ボタン2、それぞれを「イベント」として独立したスレッドとする。（シーケンス図のみに登場）
- ICトレーナ上のLEDをどのように表示させるかを記述する。
- ICトレーナ上のLEDの表示の変化を、ステートチャート図で表現する。
- この動作を実現する記述を、状態遷移表やシーケンス図で表現する。（どちらでも、記述しやすい方、または両方）

状態遷移表作成の準備

- イベント定義
  - Btn1(ボタン1が押された)
  - Btn2(ボタン2が押された)
  - Timer(spanの時間が経過した)

:

- 状態定義
  - State1(LED1～5が、2進数を表示している)
  - State2(LED1～5が、全体消灯している)

:

- 制御変数
  - led：LED1～5に表示する2進数
  - span：LEDを点滅する周期

状態遷移表

	State1	State2
Btn1	led ← led + 1 ledの値をoutする。	led ← led + 1
	⇒ State 1	⇒ State 2
Btn2	span ← span / 2 一定値以下になったら 初期値に戻す	span ← span / 2 一定値以下になったら 初期値に戻す
	⇒ State 1	⇒ State 2
Timer	消灯する	ledの値をoutする。
	⇒ State 2	⇒ State 1

報告

- 作成したいプログラムの動作を、「言葉」でできるだけ細かく記述する。（要求仕様の文書化）
- 作成したプログラムの、メインモジュールのステートチャート図を作成する。
- 状態遷移表を作成する。
- シーケンス図を作成する。
- プログラムの報告（前回と同じで構わない）
- 動作を報告する（前回と同じで構わない）
- 今回は、プログラムを記述するそれぞれの図を報告して下さい。

次回(こそ)の予告

- ライントレースカーを組み上げて、接続のための準備を行います。
- XK-1とライントレースカーが接続できることを試します。
- (この「次回の予告」ページは、予告なしに改変する場合があります。)