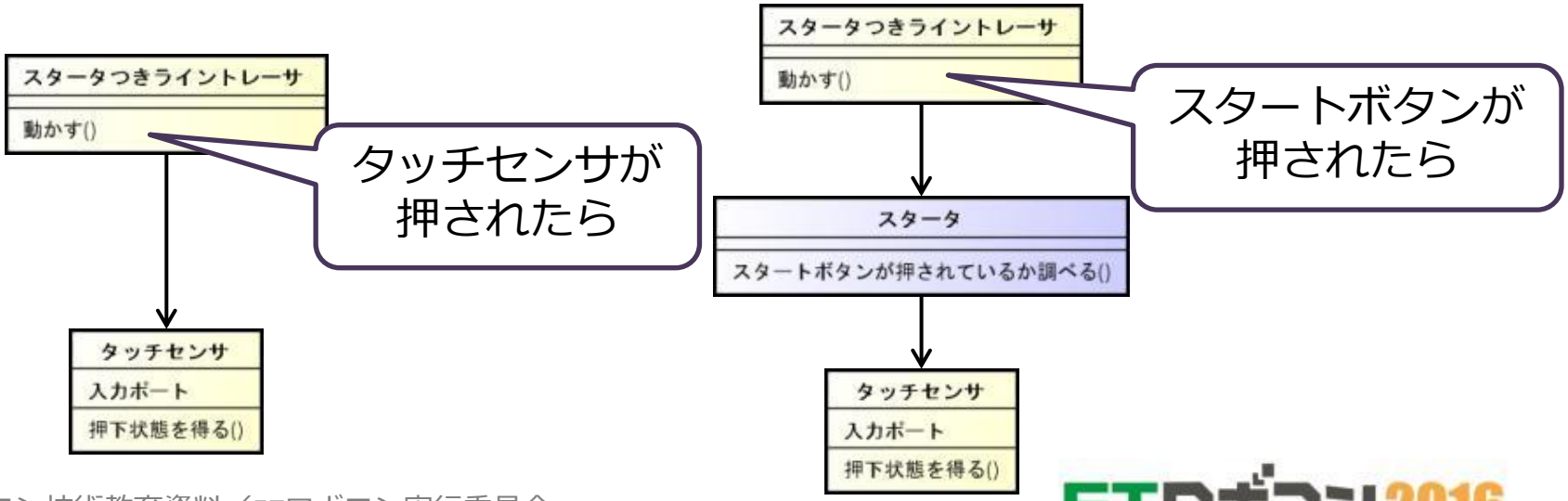
- 「モデル」についての理解は深まりましたか？
- 本日学んだことを、ぜひ大会のモデル開発に活かしてみてください
- 最後に、どのようなことを学んだのか、簡単にまとめておきましょう

9. 演習のまとめ

9-1. 役割に応じた部品を作る

- システムへの要求は大きく2種類ありました
  - 機能要求：求められた機能を提供する
  - 非機能要求：与えられたり定めた環境や材料で、性能や制約を満たす
- 機能を提供するシステムを部品の組合わせで構成しました
  - スタート指示を受付けるためのスタータを作った
  - スタータつきライントレーサは、スタータとライントレーサで構成した
- 役割に応じて部品の名前や提供する操作を考えました

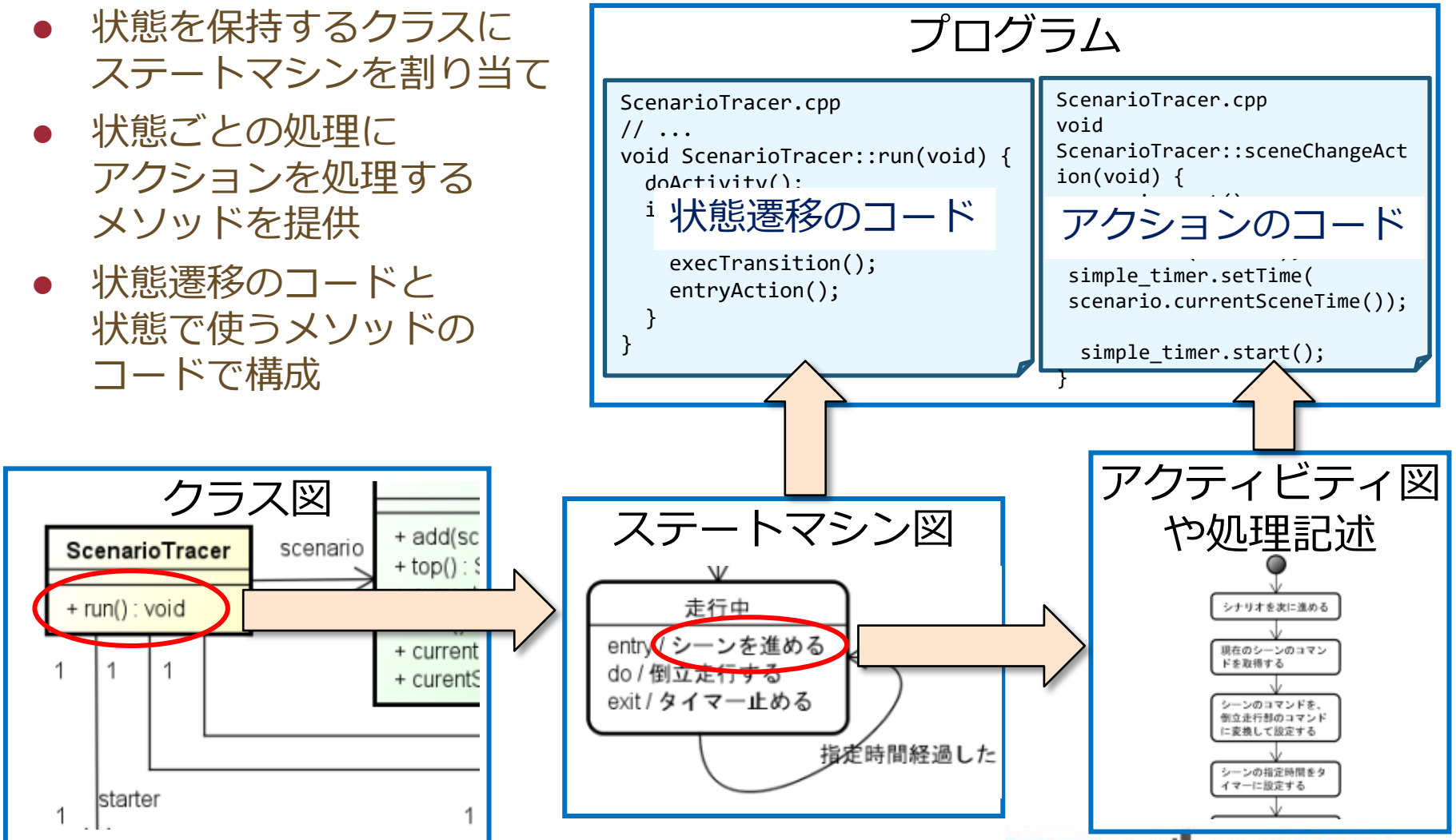


9-2. 部品を再利用する

- 部品は、別の場面（他のシステムや機能）で再利用できます
  - スタータつきライントレーサで、スタータを独立した部品にしました
  - スタータは、スタータつきシナリオトレーサやランダムウォーカーで再利用できました
    - もしスタータが違うセンサを使うことになっても他のクラスへの影響がありません
  - シンプルタイマーは、シナリオを進めるためだけでなく、ランダムウォーカーの走行方式の切り替えにも使えました
    - 同じシステムの中の別の場所に再利用できました
- システムを部品の集まりで構成すると起きる変化
  - 再利用できる部品が蓄えられるようになります
  - 部品に分けずに作る場合よりも、変更の影響範囲が限定されます
  - システムを設計するとき、部品に分けて組合わせるという考え方になります
  - 担当分けも変化が生まれます
    - △：全部設計する人＋全部実装する人
    - ○：部品を設計実装する人＋部品を使った機能を設計実装する人

9-3. 複数のモデル間につながりで作る

- 演習では、次のようなつながりを作って使いました
  - 状態を保持するクラスに ステートマシンを割り当て
  - 状態ごとの処理に アクションを処理する メソッドを提供
  - 状態遷移のコードと 状態で使うメソッドの コードで構成



9-4. ソースコードはモデルに合わせる

- ソースコードは、書き方のルールを決めて書きました
  - コード上のクラスは、実装モデルのクラスに対応させました
  - コード上のインスタンスは、実装モデルのオブジェクトに対応させました
    - オブジェクト間のリンク（関連のインスタンス）も対応させました
  - コード上のメソッドは、実装モデルのステートマシン図やアクティビティ図に対応させました
- 開発の中でのモデルの役割が高まります
  - モデルとソースコードが対応していると、モデルを使った議論がやりやすくなります
  - 変更の影響や問題の対策をモデルの上で考えられるようになります

- 「モデル」についての理解は深まりましたか？
- 本日学んだことを、ぜひ大会のモデル開発に活かしてみてください

おつかれさまでした！

ETロボコン公式トレーニング モデリング入門

モデルを使った開発を体験しよう！

－　おしまい　－