



Modelling Business Processes

Examining Process Flow

Objectives

- Familiarisation with the Activity Diagram

Contents

- Activities
- Transitions
- Decision Activities
- Object flow
- Parallelism
- Swimlanes

Activity Diagrams

- Activity Diagrams model a process life history
- Activity Diagrams model dynamic aspects of a business, a system or part of a system
 - To examine the required process flow(s) of the business
 - To identify use cases as part of a business process
 - To analyse or illustrate a use case
 - To highlight decision making

Activity diagrams are often used to document what is known about the current or envisaged business process which surrounds the system under consideration and possibly others. This can be useful to establish the overall system context especially when the analysts are not fully conversant with the business.


The activity diagrams can highlight the decision making process, and effectively show how the use cases link together, their sequence and any iterations. In this, they are a useful accompaniment to the use case diagram, which is essentially static in nature (ie has no time axis).

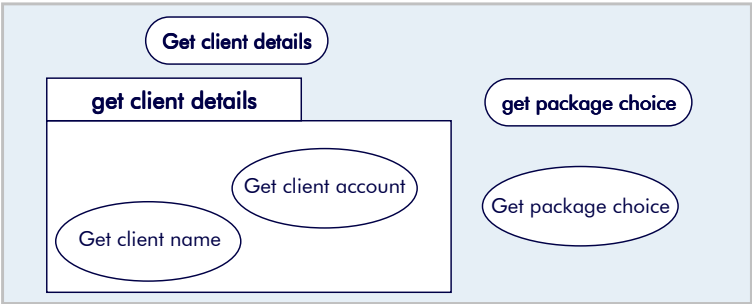
Activity diagrams can also be used within the system boundary to document how a use case flows. A use case has many scenarios, each of which is distinct from the rest because of decisions or conditions either of an actor or within the system. The activity diagram can model the smaller processes within the system (operations) and the decision points which cause different flows.

Activities

tecademy

- ▶ **An activity is a task or process**
 - ▶ **Could map to:**
 - ▶ a use case
 - ▶ a package of use cases
 - ▶ a section within a use case
 - ▶ a group of operations
 - ▶ an operation
 - ▶ **May also map to a business process outside the system**





©tecademy 4

The scope of activities is not predefined by the UML. Activities can range from whole business processes down to individual operations on objects.

An activity is represented by a *cartouche* – the pharmacist’s name for a pill of the same shape. Less formal names include ‘activity bubble’ or ‘sausage’.

The most common uses of activity diagrams are to illustrate use case flow or describe business processes.

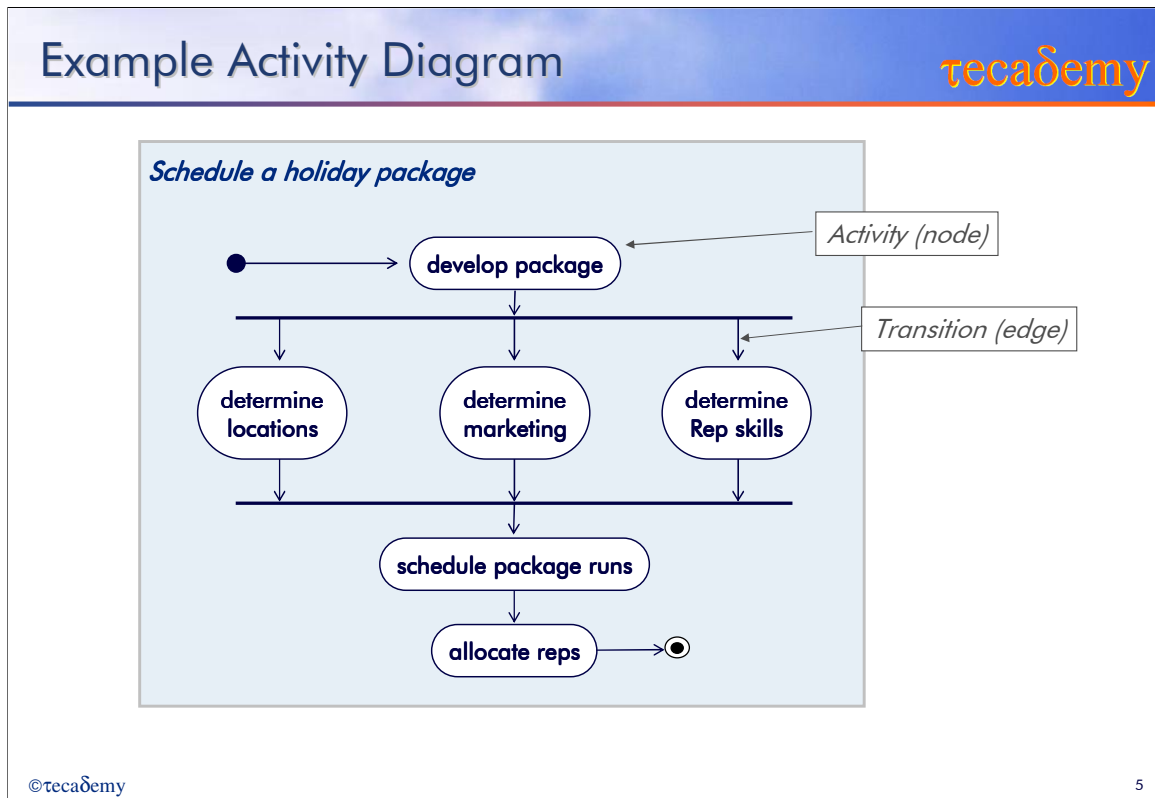
An activity could map to a single use case. Above ‘get package choice’ is a use case included by others, and is simple enough not to require siblings.

An activity could map to a business process outside the system. An example may be to phone the customer to check that they have the appropriate visa for a trip.

An activity could map to a use case package. Above ‘get client details’ may be too complex and have too many alternative possibilities to be written as a single use case. The package exists solely to keep all the related use cases together.

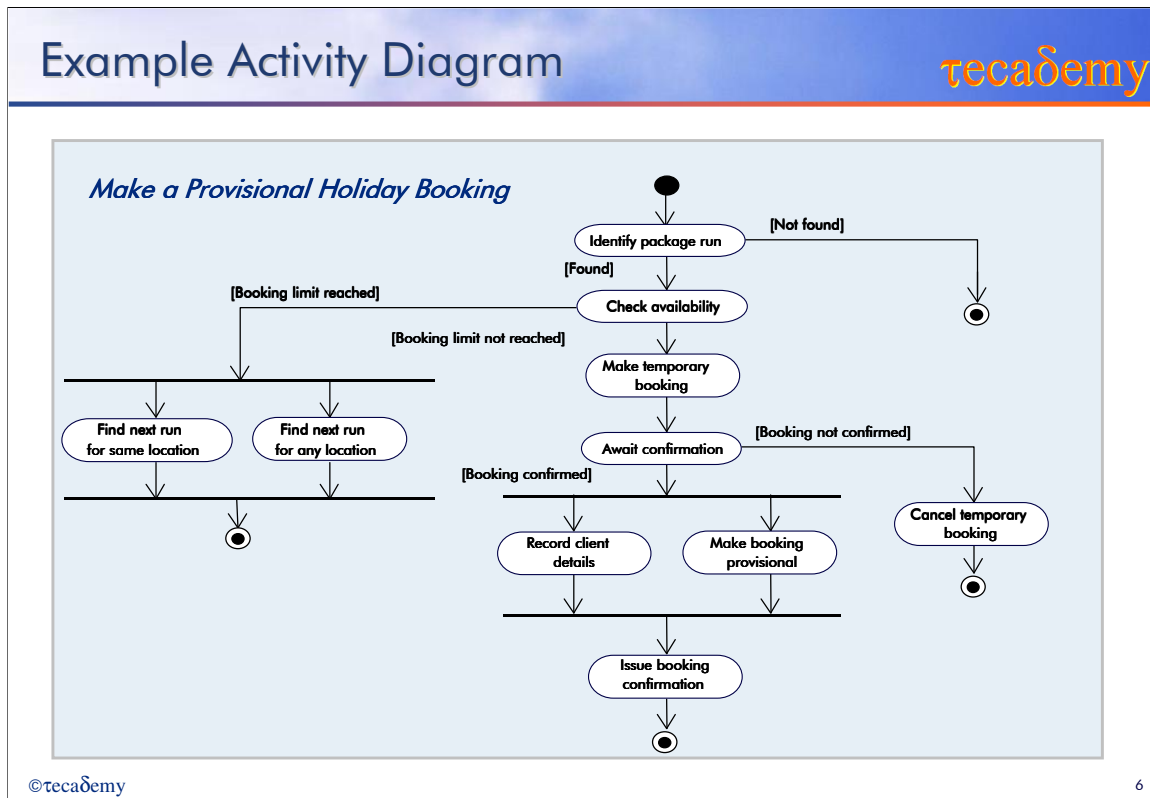
An activity could map to an operation. Above ‘find specific package run’ is an internal system behaviour to find a package run object given certain criteria.

In extremis, an activity could map to an individual line of code within an algorithm.



This is an example of an activity diagram showing a simple business process. To schedule a holiday package, first we have to develop one. When that is finished, we determine the location, determine marketing and determine the skill-set required by tour reps. These three tasks can be carried out in parallel or sequentially in any order. When all three tasks have completed, we can schedule package runs. When this has been done, we can allocate the tour reps to each run.

The 'blob' and the 'bullseye' represent the start and end points of the process. Formally, they represent the initial and final states of the process and can map to the pre- and post-conditions for the process. They can be replaced by state symbols (rectangles with rounded corners) to represent pre- and post-conditions explicitly. These can serve to verify the success of a process or the cause of failure.



Activity diagrams can be used to summarise the main and alternate flows through a particular use case. This can be a useful addition to a completed use case template since it is not easy to show conditional logic (e.g. if-then decisions) in the narrative text used to describe the different flows.

This diagram illustrates the use case text from the Make a Provisional Holiday Booking use case in the previous chapter. To book a place for a customer on a training course. When an activity diagram is mapped 1:1 to a use case, the diagrams are useful for adding a temporal dimension to an otherwise static use case model.

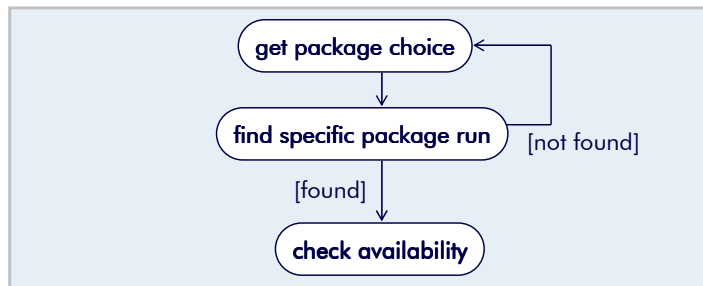
If pre- and postconditions are used in formal pairs, then the diagram can be tested to ensure that it passes the pre/post condition sets. Furthermore, it can be verified by:

- Looking for guard conditions/decision points which match preconditions
- Looking for activities that deliver the results as requested by the post-conditions
- You may well also have a different terminal activity (the bulls eye symbol) for each set of pre/post conditions.

UML allows the 'nesting' of activity diagrams within a single activity 'bubble'. This can be useful if, for instance, one needed to 'zoom in' on a single activity within a business process to see the logic within the activity. One could, in theory, 'nest' to the point where one could start at the top-level business process and zoom down through the levels to see the logic of the algorithms used in the final programming code!

Transitions

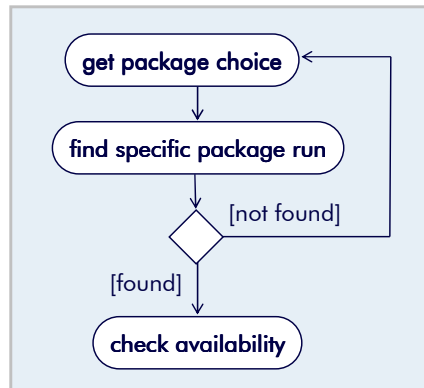
- Transitions are the flow from activity to activity
 - Completion of one activity triggers transition to next



When the activity is finished or aborted the transition may be followed. Many transitions have no label and are simply followed at the end of the activity. Transitions with guard conditions are followed at the end of the activity if and only if the condition is true. The guards on an activity should be mutually exclusive. That is there should never be two transitions which can be followed. When the guarded transitions follow an activity this means that the decision is part of the activity itself.

Explicit Decision

- An explicit decision can be shown
- Activity does not include decision
 - can be reused alone


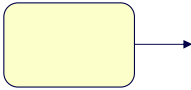
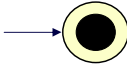
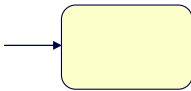
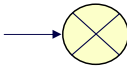


In this diagram, the activity does not include the decision and can be re-used without it.

The diamond notation can be used in two ways. One use is to show alternate paths and the condition for following one of them. If not obvious from the context, the test can be written next to the diamond. For instance, in the diagram above, the test is '*has the specified package been found?*' Adding this would add little to the information content of the diagram.

The second use of the diamond is to join transition flows. For instance, when two or more optional paths rejoin the main flow of the process.

Start and End of Process
tecademy

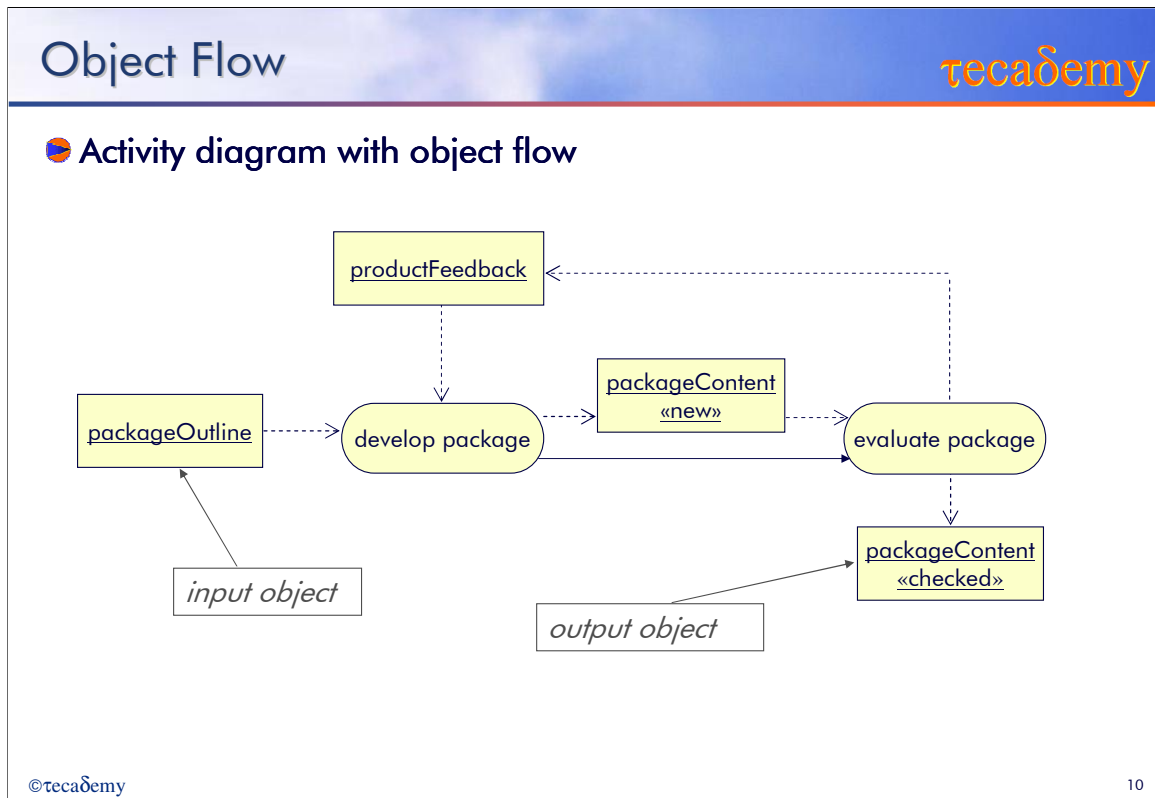
	<ul style="list-style-type: none"> ▶ Start of process <li style="padding-left: 20px;">▶ Usually only appears once per diagram
<p><i>or:</i></p> 	<ul style="list-style-type: none"> ▶ Start of process <li style="padding-left: 20px;">▶ Shown as a state <li style="padding-left: 20px;">▶ Maps to precondition
<p><i>or:</i></p> 	<ul style="list-style-type: none"> ▶ End of process <li style="padding-left: 20px;">▶ Terminates whole process <li style="padding-left: 20px;">▶ Could occur more than once per diagram
	<ul style="list-style-type: none"> ▶ End of process <li style="padding-left: 20px;">▶ Shown as a state <li style="padding-left: 20px;">▶ Maps to a postcondition
	<ul style="list-style-type: none"> ▶ End of thread

©tecademy
9

A process should have a start and an end, and this needs to be reflected in the activity diagram that models the process. The notation for process initiation is the filled-in black blob. Typically, this will appear only once on a diagram. An alternative is to show the initial state of the system at the time of process initiation. If using an activity diagram to model a use case flow, for example, this initial state would represent the precondition for that use case. In this case a state 'roundtangle' can be used containing the state definition.

The point at which a process ends can be represented by the 'bull's eye' notation. A process might contain conditional logic causing many alternative paths to termination, and so the diagram might show more than one bull's eye. Again, the final state could be shown to represent a postcondition for each of the possible use case flows.

Care should be used with the termination notation, particularly when different process threads are shown. A bull's eye ends the *whole* process, not only the thread on which it appears.



Object flow is a technique used to capture how objects participate in activities and how they are affected by the activities.

In the diagram above, a course outline is produced and used as input to the course development activity, which produces the course content as output. The course content is evaluated and the evaluation results in a checked course or feedback into the development process.

Object flows are represented by a dashed arrow. If the arrow points at an activity it is an input object; otherwise it is an output object. Object flows can be shown alongside transitions.

Parallelism

tecademy

- ▶ **A synchronisation bar allows parallel activities**
 - ▶ **A flow can fork or join**

```

graph TD
    Start(( )) --> A[develop package]
    A --> ForkBar[ ]
    ForkBar --> B(determine locations)
    ForkBar --> C(determine marketing)
    ForkBar --> D(determine Rep skills)
    B --> JoinBar[ ]
    C --> JoinBar
    D --> JoinBar
    JoinBar --> E[schedule package runs]
  
```

These can occur in parallel or sequentially in any order

This activity can only start when all three previous activities have completed

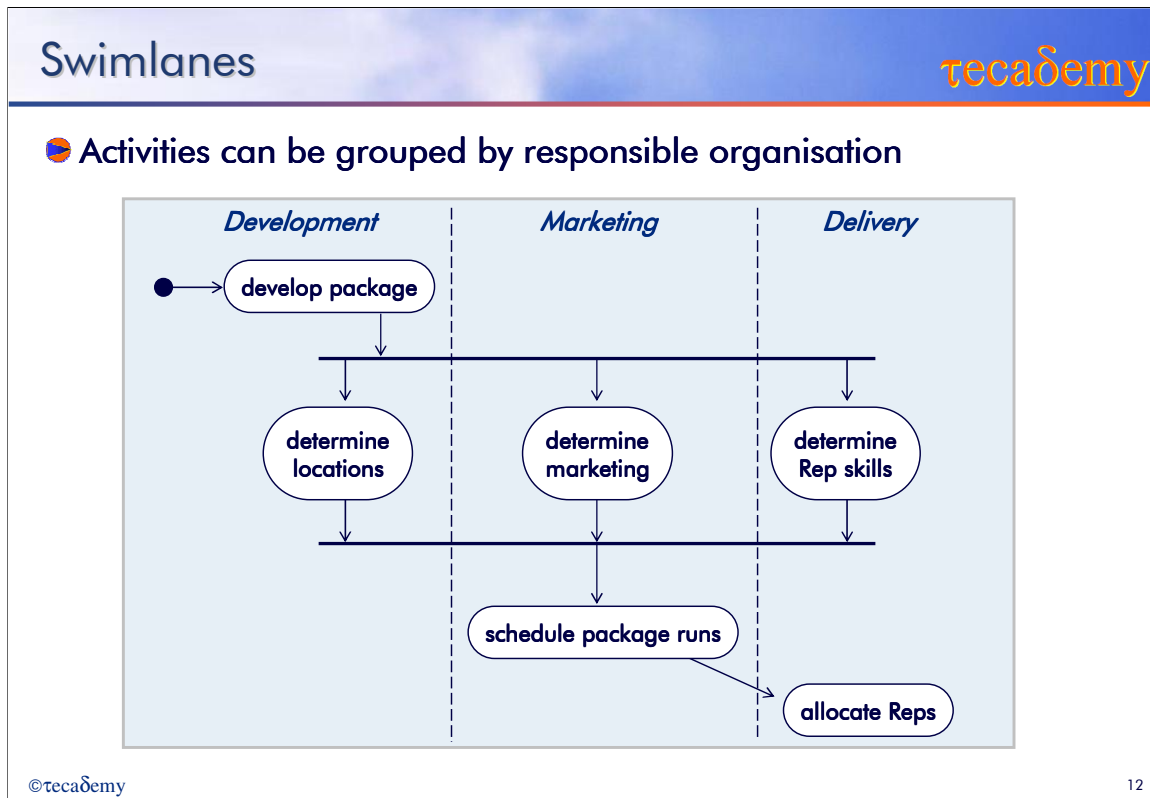
©tecademy 11

The synchronisation bar may be used as a fork or a join.

Used as a fork, it can have one incoming transition and several outgoing transitions. In this form we are specifying that the later activities have independent lifetimes - they could happen in parallel, or happen sequentially in any order, or any combination thereof (two in parallel, then a third later).

Used as a join, it can have several incoming transitions and one outgoing transition. In this form we are specifying that the outgoing transition can only be followed if all preceding activities have completed.

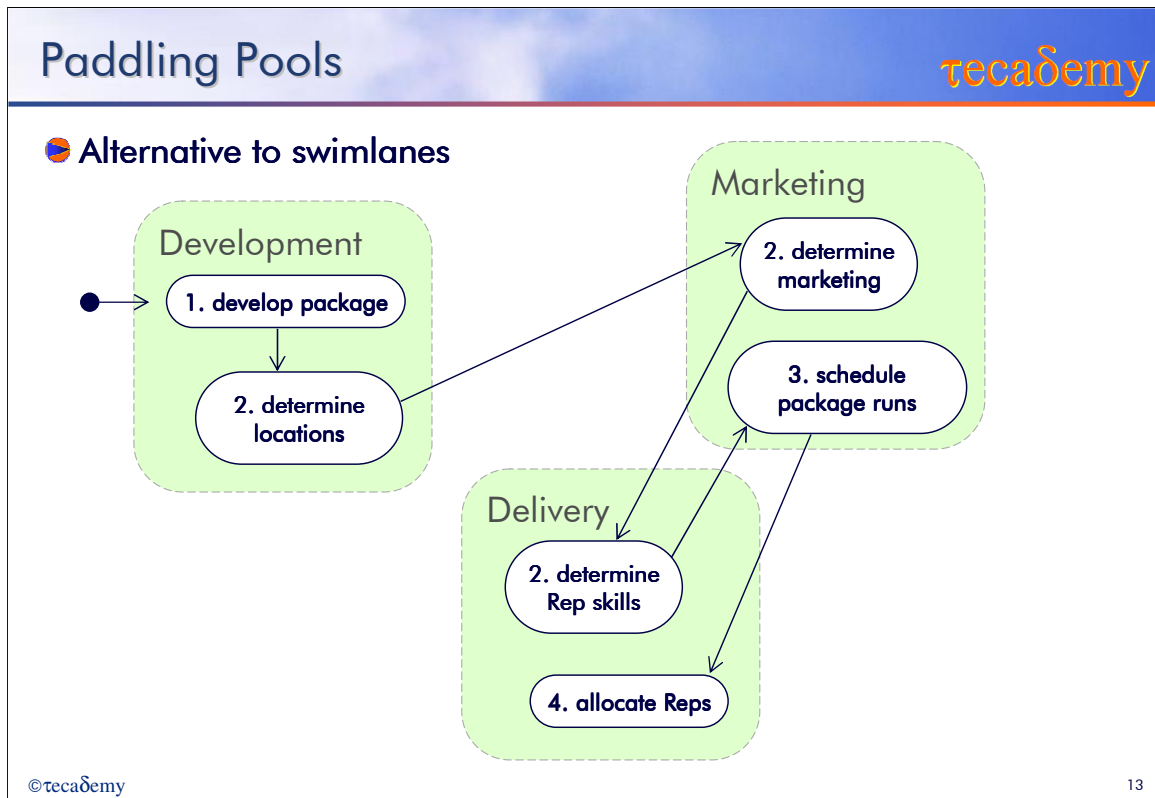
Forks and joins are normally used symmetrically, that is for every fork there should be a corresponding join with the same number of transitions.



Swimlanes can be added to an activity diagram to try and apportion responsibilities for the activities. Depending on the level of the activities (from business process to internal operations), the responsible entity could range from a company department to a class within the system.

If one of the swimlanes is used to represent the system under discussion, then it should be possible to map the activities within that swimlane to use cases within that system's boundary. Adjoining swimlanes might then map to actors for that system. For example, in the slide above, if we were exploring the requirements for a Marketing system, then 'Determine Marketing' and 'Schedule Package Runs' might be considered to be use cases of that system. Because the 'Development' role provides a trigger for those use cases, it might be regarded as a primary actor; the 'Delivery' role, which relies on a trigger from the Marketing system, could be considered a secondary actor.

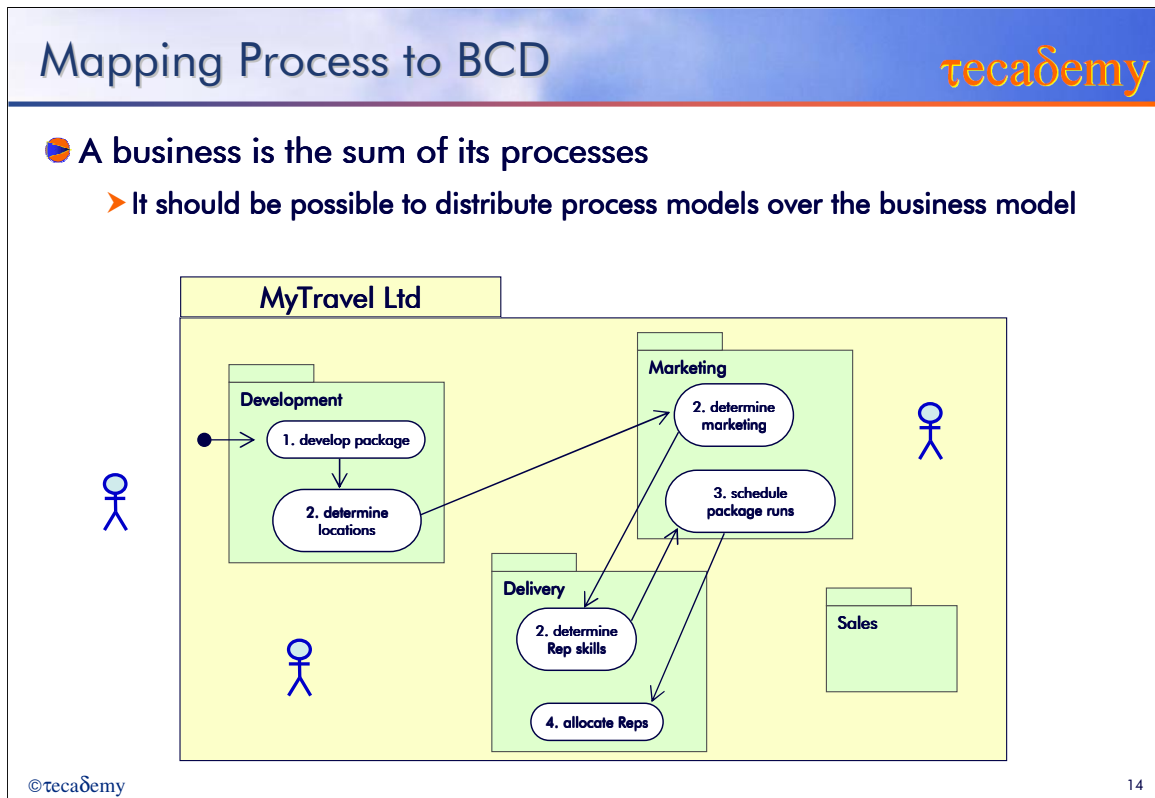
Swimlanes can also be used to show different alternative paths.



'Paddling Pools' are an informal alternative to swimlanes. They do exactly the same job as swimlanes, in that they allow the modeller to allocate responsibility for activities to individual roles who, in this case, are represented by disjoint areas – the 'pools'. Think of swimlanes as being long paddling pools arranged side-by-side.

One of the disadvantages of paddling pools is that it can make the tidy use of synchronisation bars difficult to manage. In the example above, which represents the same process as the one on the previous page, sequence numbers have been used to show activities that can execute concurrently or in any order.

One of the useful features of paddling pools can be that individual pools can correspond to the functional components of a business (represented by the packages in a Business Context Diagram (BCD)). Since the BCD represents the business, paddling pools provide a route to showing how business processes map to the Business Context Diagram (BCD), as explained overleaf...



The BCD we saw earlier was used to model the business problem within the wider context of the business and its constituent parts. Then, it was used to identify stakeholders, actors, use cases and the dependencies between them. By aligning paddling pools with the packages we used earlier to represent the different functional areas of the business, we can show how the processes we are modelling are distributed across the whole organisation and those in communication with it.

This exercise can be useful in validating our understanding of both the business and its processes. If we cannot map every process of interest to the BCD then our understanding of one or both must be at fault.

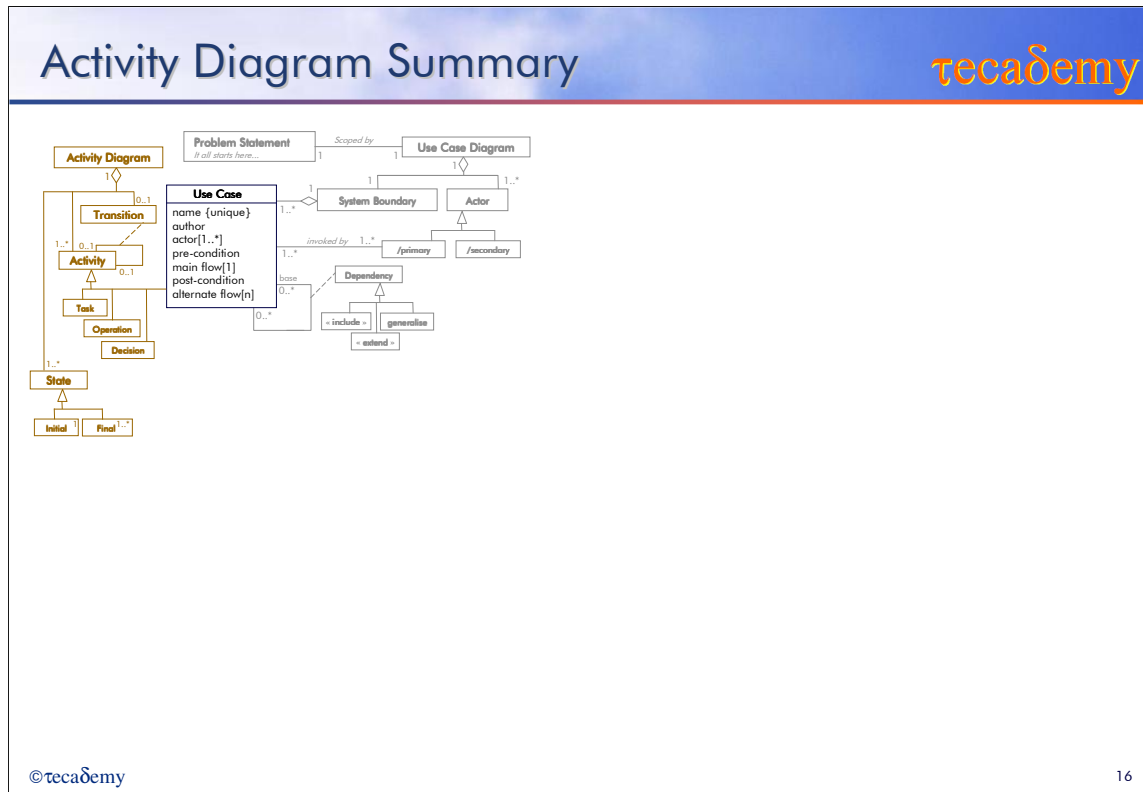
This technique can also be used to discover use cases. If we have identified our scope, or 'domain of interest', by drawing a red line around the appropriate package or packages, then it makes sense to conclude that there will be some correlation between the activities within the red line and the use cases within our system boundary.

Taken to its logical conclusion, we could develop this model to the point where every business process is identified, represented and mapped to every use case in the business. A business is the sum of its processes, and it is the processes that define the business. The use cases should support the processes and, conversely, the processes provide the context for the use cases.

It might be useful to discuss with colleagues the implications of trying to solve business problems *without* this understanding.

Summary

- An Activity Diagram models a process life history
- Activity diagrams model
 - The required process flow of the business
 - Business and process decisions
 - Use case details
- An activity is a task or process
 - Large or small
 - Inside or outside the system boundary
- A business is the sum of its processes.



A use case diagram is *static* – there is no indication that one use case follows or precedes another, except perhaps in the case of inclusions or extensions. *Activity diagrams* are *dynamic* and used to show the temporal flow of the activities that comprise an overall process – in other words, an activity diagram represents a process’ life history. A use case can be viewed as an activity, but an activity can be finer- or coarser-grained than a use case – for instance, an operation on a class can be modelled as an activity, as can a task or business process. Completion of one activity triggers a transition to the next.

Case Study – Exercise 3

- Objective

- To practice drawing Activity Diagrams.

- Turn to and complete exercise 3 in the exercise booklet.

Activity Diagrams

Notes: