**Когда надо применять DTO?**

1. DTO модели, это всего лишь прослойка между сервисами и их предназначение перенести данные от одного сервиса/слоя к другому. Это всегда чистые POCO модели. Если Вы из одного сервиса сразу будете передавать данные в другой сервис, то у Вас выйдет зависимость от типа данных. Т.е., если в первом сервисе изменится тип отдаваемых данных, то придется менять тип входных данных второго сервиса и возможно что-то даже править в коде.  
  
Прослойка из DTO моделей от этого защищает. Если в первом сервисе изменится тип отдаваемых данных, то Вы просто меняете правила мапинга отдаваемых данных на DTO модель и во втором сервисе ничего править не придется. Тоже самое касается и в обратную сторону. Ну и плюс, с точки зрения философии, позволяет эффективнее работать с моделями на уровне предметной области самого сервиса, не зависеть от моделей предметной области стороннего сервиса.

Конечно же DTO слой добавляет сложности проекту. Во первых лишний слой, лишние классы, во вторых нужны правила мапинга от одного сервиса к DTO и от DTO ко второму сервису. Поэтому это все оправдывается на крупных долгоживущих проектах с большим ожиданием изменений.  
  
Собственно по поводу конкретики применения я ответил: перед тем как отдать данные - мапите в DTO, получили DTO - сразу мапите в локальные модели. Больше внутри нигде их не используете, они только переносят данные.

2. Фабрика нужна там, где нужен контроль создания объекта. Либо ограничение прав его создания, либо принятие решения создания по условию и.т.д.

Я так понимаю, что DTO подойдет, в случае если в итоговом сервисе я просто “скопом” вывожу клиенту все данные, которые я получил от первого сервиса или от ДТО-класса. А в случае если в итоговом сервисе эти данные (перед отправкой клиенту) обрабатываю не “скопом”, а по одному, то чтобы не менять код два раза (и в DTO и в итоговом сервисе), лучше обходиться без DTO.