Key Strategies for Object Identification 1. Underline the Noun Strategy Works directly with the written problem. Underline each noun or noun pharse and trat it as a potential object. Identification can be put into different acetgories: - Objects of interest - Classes of interets - Actors - Unitresting objects - Atrributes - Events

2. Identifiy the Causal Agents Every effect is casued by an element. This startegy looks to identifiy the cause things to happen. - Produce or control actions - Produce or analyze data - Provide interfaces to people or devices - Store information - Provide services to people or devices - Contain more fundamental objects as parts A causal object is an object that autonomously performs actions, coordinates the activities of component parts, or generates events

3. Identify Services The services will be provided by objects. Provide passive control: do what is requested Data storage or both. For example: switch is a passive control object becasue does not initiate action It provides services to the casual objects (turns the ligth on or off) Also known as servers. (provide services to clients) - does not know their clients

4. Identify Messages and Information Flows For each message: sender, reciver (potenitally a processer) Correspond: information and control flows Relized by: operation calls or receptions The message can be an object too. Remote communcation with network packets. (remberd or processed)

5. Identify Real-World Items

1 Problem 5.1 Apply Nouns and Causal Agents Strategies

In this problem we will limit ourselves to applying the “underline the nouns” and identify causal agents” strategies. Those are the most common but least helpful strategies. Application: read the parts of the problem statement that apply to the use case in question, and then underline the nouns and noun phrases. It only finds the explicitly stated objects and also identifies attributes, actors, objects we don’t care about, and synonyms for already identified elements. The causal agent strategy looks for autonomous and triggered behaviours in the system and tries to identify the sources and targets of those behaviours.

1.1 Roadrunner Traffic Light Control System

Three types detector supported: SLIP (subsurface passive loop inductors), AIS (above-surface infra-red sensors), ASR (above surface radars)

SPLI uses wired interface to communicate with the controller, ASI, ASR use both wired and secured wireless. Detectors can count the vehicles.
ASI and ASR are able to receive directional transmission priority and emergency vehicle transmitters. The maximum range of such reception is between 250 and 1000 feet.
When a vehicle enters the detection area, the detector report the presence of a vehicle. Separate detectors are used for each lane in each direction.

1.1.1 Mode 2: Fixed Cycle Time

The most common operational mode. In this mode, the lanes cycle GREEN-YELLOW-RED in opposite sequences with fixed intervals. Must ensure: If any traffic light is non-RED, then all the lights for cross traffic must be RED and pedestrian light must be set DO NOT WALK. The default values depend on the system configuration. Turn lane and pedestrination times..

When a problem is bigger like the CUAV, it is common to apply the object identification strategies at the subsystem level. This sepcification includes the subsystem use cases, the system architecture into which the subsystem must fit and complate sepcification of the interfaces the subsystem must provide or may require from other elements.

Reconnaissance Management use cases.

1.1.2 The Unmanned Air Vehicle (UAV)

It operates at an altitude of up to 30,000 feet with ground speeds of up to 100 knots (cruise) and 150 knots (dash) and carry a payload of up to 450 lbs for durations in excess of 24 hours. Files unimpeded in low-visibility environments while carrying either reconnaissance or attack payloads.

1.1.3 The Coyote Mission Planning and Control System (CMPCS)

Mobile CMPCS with capability to control up to four UAVs with a manned control station per UAV that fits into a smaller towable trailer.

Mission Modes
Normal mission modes include:

- Preplanned reconnaissance
- Remote-controlled reconnaissance

- Area search
- Route search
- Orbit point target
- Attack

The CMPCS is housed in a 30 x 8 x 8 triple-axis trailer that contains stations for pilot and payload operations, mission planning, data exploitation, communications, and SAR viewing.

2 Problem 5.2 Apply Services and Messages Strategies

The Services strategy looks for the services or operational contracts that the system containing the collaboration provides. Each of these services must either be met by a single object or be decomposed into nested services, each of which must be provided by a single object. The Messages strategy applies the same principle to messages and information flows: every message or information flow must be either provided or received by an object in the collaboration.

3 Problem 5.3 Apply Real-World Items and Physical Devices Strategies

These two strategies are related in the sense that they seek to identify objects and classes that represent things that exist in the physical world. The real-world items strategy seeks to identify objects from the real world that have information or resource that must be managed int the system. For instance: in a banking system, the customer is clearly an object in the real world and we must maintain about information. (Name, Adress, Tax, etc).

The Physical Devices Strategies identifies physical devices which are part of the system. The best approach is not to model all phsyical devices that the system must monitor and control as actors.

4 Problem 5.4 Apply Key Concepts and Transaction Strategies

The Key Concepts strategy is the antipode of the real-world items strategy. It seeks to find the essential concepts of a domain of discourse, particularly when these elements are abstractions and have no physical manifestation.

A transaction is the reification of an interaction (among objects) into an object itself. A “withdrawal” from a bank account is a transactional object because it must be remembered for a period of time and reported to the customer for account reconciliation. A request for an elevator to go to a particular floor is also a transactional object because it must be remembered until the elevator actually arrives at the floor.

5 Problem 5.5 Apply Identify Visual Elements and Scenarios Strategies

In systems with nontrivial user interfaces, these strategies often work well together, because one of the issues that arises when considering scenarios that involve the human users of the system is how the user interfaces gather and provide information to the internal parts of the system that must deal with and respond to that information.
Scenario strategy the easiest way to construct a verifiably working collaboration of classes to realize a use case.
Simply start with a use-case scenario. The use case will have a (possibly large) number of scenarios as exemplars, illustrating examples of the use case unfolding as specific messages or events come in to the system. We will simply elaborate object roles to show how the scenario unfolds at the object, rather than the system, level.

This elaboration can be done “in-line” or by decomposition. By “in-line,” I mean that I copy the original scenario and start adding object lifelines to the copy. The decomposition approach is done by decomposing the lifeline into a more detailed scenario, a feature added in UML 2.0 and supported by Rhapsody.

When I discover that I need to have a service performed somewhere inside the system, I ask myself the following questions:

- What object has the information necessary to perform this service?
- What object has the proper interfaces necessary to perform this service?
- What object has the responsibility to perform this service?

I add objects, services, and relations to detail this scenario. I may start at the beginning of the scenario, or somewhere in the middle—perhaps at a very important part of the use case or some part that I feel I have a good handle on.

At first glance, the Detect Vehicle use case doesn't seem to be an ideal candidate for this strategy. After all, the "scenario" consists of a single message "vehicleDetect" from the vehicle to the system (well, almost anyway). Each of the detectors must be configured differently.

The radar detector is the only active emitter. It only emits a specific frequency but both the strength of the emitter and the sensitivity of the detector can be set.

Each of these scenarios references another—one that returns statistics to the user, including average vehicle count per hour and total count since reset. This latter scenario is referenced by all three scenarios because the generation of statistics depends only upon the detection of a vehicle and not how they are detected.

The astute reader will note that the inclusion of the formal parameter flowID, along with the diagrammatic comment describing the effects of the different values of this parameter, is just a sneaky way of putting many different physical scenarios on a single sequence diagram, as is the use of the unordered interaction operator.

As far as the human interface goes, the external UI has already been planned out and provided in the requirements specification.

6 Problem 5.6 Merge Models from the Various Strategies

In this last problem of the chapter, you must merge together the different models that have arisen from the application of the different strategies. In actual use, the models would be most likely incrementally constructed by taking what's already there and adding objects, classes, relations, and features identified in additional strategies.

For the Roadrunner Traffic Light Control system, we've worked, in part, on two different use cases, Fixed Cycle Time Mode and Detect Vehicle. The system includes the elements that collaborate to realize both these use cases. These use-case collaborations have been shown as two independent class diagrams and they will continue to be shown in that way. Why?
The reason is that to scale the use of modeling techniques to real-world problems, we must have some criteria for deciding what goes on which diagrams.

The CUAV is a larger scale system, so we've narrowed our focus onto a single use case (Acquire Image) for a specific subsystem (Reconnaissance Management). One can imagine teams of people working independently, possibly in different areas of the world, to develop the use cases for their own subsystems.

Finally, for both sample problems, answer the following questions:

- Which strategy worked best for you, and why?
- Which strategy worked the least well for you, and why?
- Did some strategies seem to be better identifying elements in one problem or the other? Can you generalize that so that you know when to apply which strategies?
- What combination of strategies do you think will be most effective for you?

7 Looking Ahead

We have used a number of different strategies to identify the "essential" or "analysis" objects and classes. These strategies are only partially orthogonal; that is, they identify some, but not all, of the same model elements.

The analysis model exactly corresponds to the idea of a platform independent model (PIM) in the OMG's model-driven architecture (MDA).It is devoid of design decisions.

While analysis is all about identifying elements required to meet the functional requirements, design is all about optimizing the analysis model to meet performance requirements and other design optimization criteria. Different optimization goals or different technology selections result in different PSMs from the same PIM.