## T1: Vehicle Cruise Controller Simulation (VCCS)

Agents in the VCCS

Agents in the system are of two kinds:

Components of the engine:
throttle_manager, engine, vehicle, speed_transducer, environment

Interface agents:
cruise_cutout, driver

Cf. object-oriented model taken from Booch / Deutsch <picture>

< background issues: stimulus-response model,
limitations of OOP model etc >

All agent instances are persistent but for the autoThrottle_manager.
The autoThrottle_manager exists only whilst the cruise controller operates.

2. Analysis of the LSD specification for agents

Issues for the designer (to be addressed by an LSD specification):

- What are the key observations of a system that explain its behaviour?
- What are the agents in the system?
- How are the observations associated with agents?
- How does an agent register changes of state in its environment?
- In what circumstances does an agent have privileges to change state?
- What observations can an agent change?

In experiment, correlate values of observations o1 and o2.
Simplest case, as illustrated by Hookes' Law: "change o1 and observe o2".
Can still recognise FD where another agent is responsible for changing o1.
There is a semantic distinction between o1 and o2:
in changing o1, refer to o1 not as an observation but as state variable
in observe o2, regard o2 as an oracle.

[A spreadsheet user may continue to update though the spreadsheet has yet to be made consistent. In effect, if o1 is displayed value of cell1 and o2 is displayed value of cell2, then the relation o2 = f(o1) is guaranteed to hold only when the spreadsheet is completely up-to-date. This can be understood in terms of the above model as: o2 is an oracle to the spreadsheet agent. In observing o2, the spreadsheet agent does not simply consult the display, but (as when following a protocol for an experimental observation) waits until the spreadsheet is up-to-date.]

The use of derivates in animation from an LSD specification reflects presumed FDs and synchronisations in the view of the system designer. This interpretation may conflict with the concept of a derivate as an indivisible relationship as perceived by an agent.

[In a model of spreadsheet and spreadsheet user system, have to appeal to the protocol for observation used to determine the value of a cell described above to account for the fact that the spreadsheet user perceives definitive relationships that in the view of the system designer are not universally valid. This is good illustration of how a v-oracle is to be interpreted with reference to subtle conventions of communication and observation. Notice also that the indivisibility of relationships between spreadsheet cells defined by formulae in the view of the spreadsheet user is here guaranteed because no other agent can interrupt the updating process - if this were not the case, the premise on which the user continues to enter data whilst the spreadsheet is still updating would be invalid.]

The usefulness of the concept of a derivate as an indivisible relationship as perceived by an agent is illustrated by the derivates used in the station_master agent in the train specification.

The LSD specification of an agent comprises 4 kinds of variable:

**oracle** - a variable to which it responds
**state** - a variable that it owns
**handle** - a variable conditionally under its control
**derivate** - an indivisibly coupled stimulus-response relation (?)

Issues

*Is variable a good name for these features of the LSD specification? In a sense, they are all observations, but in some instances they are observations associated with **different** agents of what are conceptually the **same** entity. Adopt v-observation to mean observation designated by identifier v.*

This is why the same identifier occurs many times in an LSD specification

| | | |
|---|---|---|
| e.g. | measSpeed | **state** for speed_transducer |
| | | **oracle** for throttle_manager |
| | | **oracle** for the driver |
| | cruiseStts | **state** for cruise_cutout |
| | | **oracle** for the driver |
| | | **oracle** for the throttle_manager |

*In simulation derived from an LSD specification, identifier instances will in general be represented by different variables.*

The system designer determines the observations recorded in the LSD specification. These observations are what the designer would record and subject to experiment in relating the (pre-act-of-faith) behaviour of the system to the (post-act-of-faith) reliable activity of the components.

Classification of variables associated with an agent

**states**

There are generally certain observations associated with an agent in such a way that were the agent instance to disappear they would also disappear. E.g. accel, windF, actSpeed, .... are meaningful only whilst there is a vehicle agent. Such observations are identified as state variables of the agent. They are the variables bound to the agent.

A state variable v is the primary source of all v-observations: at any time, all v-observations are associated with at most one state variable.

A state variable v is a record of the "authentic value" of v-observations. It's value is not necessarily the value perceived by the agent to which it is bound. That is to say, a variable may be both an oracle and a state to the same agent, as in "The time as recorded by my watch, or my bank balance".

Note that those agents which serve primarily as observers and actors in a system tend to have few state variables. E.g. there are no state variables associated with the driver agent.

**consts**

Constants are a particular kind of state variable, whose value is not subject to change through actions of agents in the simulation e.g. the physical parameters of the vehicle: mass, windK, rollK.

In the VCCS as implemented, it would be possible to treat the parameters of the vehicle as subject to change. E.g. a simulation might take account of loading the vehicle, or express the relationship between the wind resistance and the vehicle profile.

**oracles**

In anthropomorphic terms: an **oracle** = value as perceived by an agent.
This admits the possibility that its value is inaccurate.

actSpeed is **state** in vehicle, ...

*Is actSpeed an oracle to the speed_transducer? In what sense is the fact that a door is locked an oracle to the*

*door user? Whether the door can be opened or not in no way depends on the user's perception of whether it is or isn't locked, nor is the effect of locking the door subject to delay.*

**handles**

A handle is a variable that an agent can manipulate, subject to certain enabling conditions being met. A handle variable of an agent is not necessarily bound to the agent. E.g. the driver agent can act to change the state variable engineStts of the engine agent.

Handles allow direct action of one agent upon another that is outside the scope of an object-oriented paradigm (cf an object invokes a method to change its state).

When a v-handle and v-oracle co-exist, they together indicate communication between agents. When animating from a specification, appropriate assumptions about communication have to be made. For instance, the VCCS specification does not specify how the speed of the vehicle, as measured by the speed_transducer, is communicated to the throttle_manager. In animation, we might assume idealised communication, or refine the specification to include further details of the communication channel.

**protocols**

An agent's privileges to redefine handles make up its protocol.

Each privilege represents a possible action, or sequence of actions, that an agent can perform in appropriate circumstances. For example, if the engine is switched off, the driver can switch it on, and vice versa.

There is no fixed set of rules for interpreting agent protocols in operational terms. Relevant considerations include:

- Privileges are not obligations to act. Many possible alternative courses of action are represented in the driver protocol; for example, when the cruise controller is switched on, the driver can switch it off, request it to maintain the vehicle either at the specified cruise speed or at the current measured speed, or reset the specified cruise speed.

- Certain privileges express actions that must be executed promptly as soon as they are enabled. For instance, the cruise controller should not try to maintain the vehicle speed if once the driver touches the brake (see the cruise_cutout protocol).

- Animation from an LSD specification should reflect the external significance of agent actions. For instance, it would be unreasonable to expect the driver to switch the cruise controller on and off rapidly and repeatedly; not only is this is an improbable activity for the driver, but there will be an engineering constraint on how fast it is possible to switch between on and off modes.

Two kinds of state-changing activity:
    observation of change in particular context
        modelled by redefinition in a definitive script
    change of context
        modelled by introducing or deleting components of the script

Animation when event-driven
    vs asynchronous delays / non-deterministic response
Pat Sockett's Philip Horgan's projects

analogue variables

are derivates more independent of agent specifications than originally conceived?