Contextual Substitutability “Car Park” Demo Application

Premek Brada <brada@kiv.zcu.cz> *1.Feb 2012*

The application consists of several bundles. It evolves in 6 application revisions between which combinations of changes to bundles happen that lead to strict and/or contextual substitutability. Descriptions here are those not clear from bundle manifests.

Abbreviations

- svc = service
- pkg = package
- v = version

Core and shared

- Namespace prefix: **cz.zcu.kiv.osgi.demo.parking** .
- All bundles except statsbase use SLF4J for logging => import package org.slf4j .
- All bundles except statsbase have activator => import org.osgi.framework

Description of application revisions

Rev 1 – Baseline

Dashboard uses GateStats on its “gate” port only through its CountingStats subset. Gate provides LaneStatistics since there is no other bundle to do that.

Rev 2 – Gate observes CarPark status

Gate adds dependency on ParkingStatus, provided by CarPark.

Rev 3 – TrafficLane added

LaneStatistics evolved by adding a method (which however is not used by clients in app rev 3-5). TrafficLane component added to the system, provides LaneStatistics but the Dashboard is still connected to the Gate which still provides LaneStatistics as well. Gate consequently also runs the traffic simulation internally. Dashboard now uses full GateStatistics on its “gate” port.

Rev 4 – TrafficLane Statistics used

TrafficLane component's interface did not change, but its LaneStatistics are now used by Dashboard, because Gate stopped providing LaneStatistics. Internally the TrafficLane implementation changed to run the traffic simulation. CarPark now provides ParkingStatistics which however are not used by anybody. StatsBase changed, now provides an abstract implementation of CountingStatistics interface, all statistics classes (gate, lane, carpark) inherit from this abstract base.

Rev 5 – RoadSign added

RoadSign component added, driven by TrafficLane; shows a sign about car park's free capacity. CarPark still provides ParkingStatistics which got changed by removing a method but which are still not used by anybody.

Rev 6 – TrafficLane Statistics modified

LaneStatistics interface evolved by replacing a method with another one; that method however was not used by anybody in previous app revisions.

Bundle revisions and pkg versions per application revisions

Unchanged are in normal weight, changed are in boldface, newly added are with asterisk.

TODO more detailed analysis of imports – what actually is used, reflect in DIFFs listed (provide both standard diff and detailed analysis diff).

Rev 1

All diffs are "NA" because there is no preceding revision.

***StatsBase** 1 = 1.0.0

- ex: statsbase;v=1.0.0
- im: (none)

***Dashboard** 1 = 1.0.0

imports statsbase because parent interface is used on "gate" stats port

- ex: (none)
- im: gate.statistics;v=[1.0.0, 2.0.0) , lane.statistics;v=[1.0.0, 2.0.0) , statsbase;v=[1.0.0, 2.0.0)

***Gate** 1 = 1.0.0

- ex: gate.statistics;v=1.0.0 , lane.statistics;v=1.0.0
- im: carpark.flow;v=[1.0.0, 2.0.0) , statsbase;v=[1.0.0, 1.1.0)

***CarPark** 1 = 1.0.0

- ex: carpark.flow;v=1.0.0 , carpark.status;v=1.0.0
- im: (system only)

Rev 2

StatsBase 1 = 1.0.0 from rev 1

Dashboard 1 = 1.0.0 from rev 1

Gate 2 = 1.0.1 (NON)

INS on imported packages (ParkingStatus)

- ex: NON gate.statistics;v=1.0.0 , NON lane.statistics;v=1.0.0
- im: INS carpark.status;v=[1.0.0, 2.0.0) , NON carpark.flow;v=[1.0.0, 2.0.0), NON statsbase;v=[1.0.0, 1.1.0)

CarPark 1 = 1.0.0 from rev 1

Rev 3

StatsBase 1 = 1.0.0 from rev 1

Dashboard 2 = 1.0.1 (NON)

SPE on imported(!) interface (CountingStats → GateStats) leading to DEL on imported pkg

- ex: (none)
- im: NON/SPE gate.statistics;v=[1.0.0, 2.0.0) , NON lane.statistics;v=[1.0.0, 2.0.0) , DEL statsbase

Gate 3 = 1.1.0 (SPE)

SPE on exported package/interface LaneStatistics, INS on exported packages (VehicleSink), internal change: spawns traffic lane simulation thread

- ex: NON gate.statistics;v=1.0.0 , SPE lane.statistics;v=1.1.0 , INS gate.vehiclesink;v=1.0.0
- im: NON carpark.flow;v=[1.0.0, 2.0.0) , NON carpark.status;v=[1.0.0, 2.0.0) , NON statsbase;v=[1.0.0, 1.1.0)

CarPark 1 = 1.0.0 from rev 1

***TrafficLane 1 = 1.0.0**

- ex: NA lane.statistics;v=1.1.0
- im: NA gate.vehiclesink;v=[1.0.0, 2.0.0) , NA statsbase;v=[1.0.0, 1.1.0)

Rev 4

StatsBase 2 = 1.0.1 (SEM-SPE)

see below for reasoning behind the version change

- ex: SEM-SPEC statsbase;v=1.0.1
- im: (none)

Dashboard 2 = 1.0.1 from rev 3

Gate 4 = 2.0.0 (MUT)

DEL on exported packages (LaneStatistics), SPE on exported pkg: iface GateStatistics adds ParkingStatus operations, SEM-SPE on exported pkg: VehicleSink adds throw exception, SPE on imported carpark.status package

- ex: SPE gate.statistics;v=1.1.0 , SEM-SPE gate.vehiclesink;v=1.0.1 , DEL lane.statistics
- im: NON carpark.flow;v=[1.0.0, 2.0.0) , SPE carpark.status;v=[1.1.0, 2.0.0) , statsbase;v=[1.0.0, 1.1.0)

CarPark 2 = 1.1.0

INS on exported packages (ParkStatistics), INS within exported package (IParkingStatusUpdate in carpark status), SPE on exported interface ParkingStatus (within carpark status), SEM-SPE within VehicleFlow implementation (adds throw exception)

- ex: SEM-SPE carpark.flow;v=1.0.1 , SPE carpark.status;v=1.1.0 , INS carpark.statistics;v=1.0.0
- im: NON statsbase;v=[1.0.1, 2.0.0)

TrafficLane 2 = 1.0.1

internal change of implementation

- ex: NON lane.statistics;v=1.1.0
- im: NON gate.vehiclesink;v=[1.0.0, 2.0.0) , SEM-SPE statsbase;v=[1.0.1, 1.1.0)

Note on the version of "statsbase" package and whole bundle: the changes introduced are NON on the API part (the ICountingStatistics iface unchanged) and INS on the implementation part (abstract base class added). These changes don't break existing clients (who only used the interface so far) neither existing providers (because the interface hasn't changed). In this fairly singular case therefore, the change should IMHO be classified as internal.

OTOH, it makes versioned dependency difficult, needing to use the 1.0.1 as lower bound which somehow feels odd. But if 1.1.0 was used as pkg version, it would render the bundle incompatible with previous revisions of other bundles in app revs 1-3 which clearly would be wrong.

Lastly: implementation of statistics classes extend the abstract base, which makes its public methods removed from their public interface; we do not classify this as DEL → GEN difference because clients will not notice the difference.

Rev 5

StatsBase 2 = 1.0.1 from rev 4

Dashboard 2 = 1.0.1 from rev 3

Gate 4 = 2.0.0 from rev 4

CarPark 3 = 2.0.0

GEN on exported package/interface ParkingStatistics (method removed)

- ex: NON carpark.flow;v=1.0.1 , NON carpark.status;v=1.1.0 , GEN carpark.statistics;v=2.0.0
- im: SEM-SPE statsbase;v=[1.0.1, 2.0.0)

TrafficLane 3 = 1.1.0

INS on imported packages (RoadSign)

- ex: NON lane.statistics;v=1.1.0
- im: NON statsbase;v=[1.0.1, 2.0.0) , NON gate.statistics;v=[1.1.0, 2.0.0) , NON gate.vehiclesink;v=[1.0.0, 2.0.0) , INS sign.roadsign;v=[1.0.0, 2.0.0)

***RoadSign 1 = 1.0.0**

- ex: sign.roadsign;v=1.0.0
- im: (system only)

Rev 6

StatsBase	2 = 1.0.1	from rev 4
Dashboard	2 = 1.0.1	from rev 3
Gate	4 = 2.0.0	from rev 4
CarPark	3 = 2.0.0	from rev 5
RoadSign	1 = 1.0.0	from rev 5

TrafficLane 4 = 2.0.0

MUT on exported package/interface LaneStatistics

- ex: MUT lane.statistics;v=2.0.0
- im: NON statsbase;v=[1.0.1, 2.0.0) , NON gate.statistics;v=[1.1.0, 2.0.0) , NON gate.vehiclesink;v=[1.0.0, 2.0.0) , NON sign.roadsign;v=[1.0.0, 2.0.0)