Guided Architecture Trade Space Exploration

Fusing Model Based Engineering and Design by Shopping

Sam Procter

Lutz Wrage

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Document Markings

Copyright 2019 Carnegie Mellon University. All Rights Reserved.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

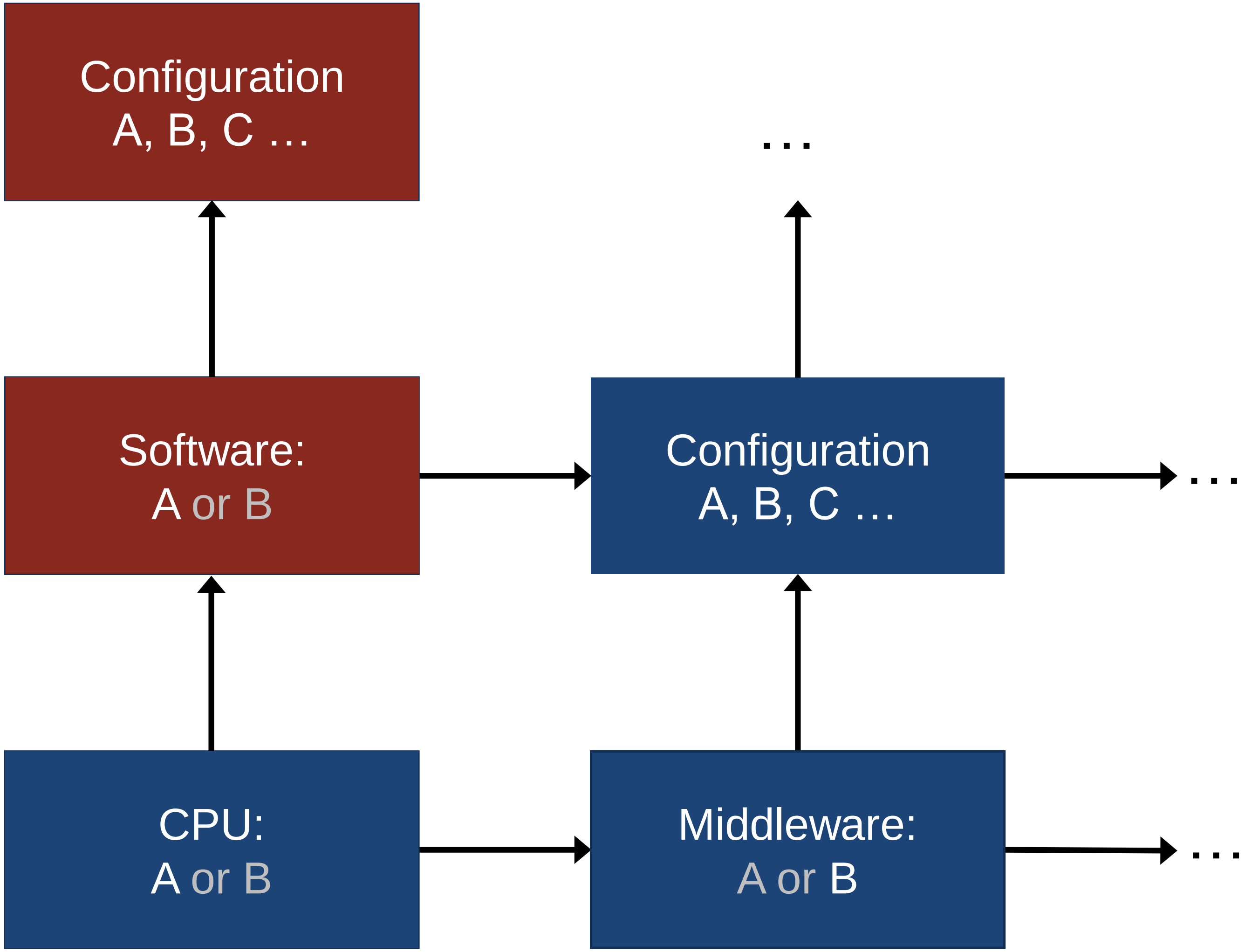
NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

DM19-0897

More components, more complexity



Configuration
A, B, C …

…

Software:
A or B

Configuration
A, B, C …

…

CPU:
A or B

Middleware:
A or B

…

© 2019 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

But that’s not actually how it all works.

System designers rely on their *expertise* and *intuition* instead

- Model-Based System Engineering (MBSE) supports that intuition, but has some drawbacks at large scale.
- Design Space Exploration works well at scale, but has some usability issues and rarely uses multipurpose system models

So, we created and evaluated the *Guided Architecture Trade Space Explorer*, which supports designers’ intuition.

© 2019 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Outline

A Wheel-Braking System

Designing by Shopping

Guided Architecture Trade Space Exploration

© 2019 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release
and unlimited distribution.

Outline

A Wheel-Braking System

Designing by Shopping

Guided Architecture Trade Space Exploration

© 2019 Carnegie Mellon University

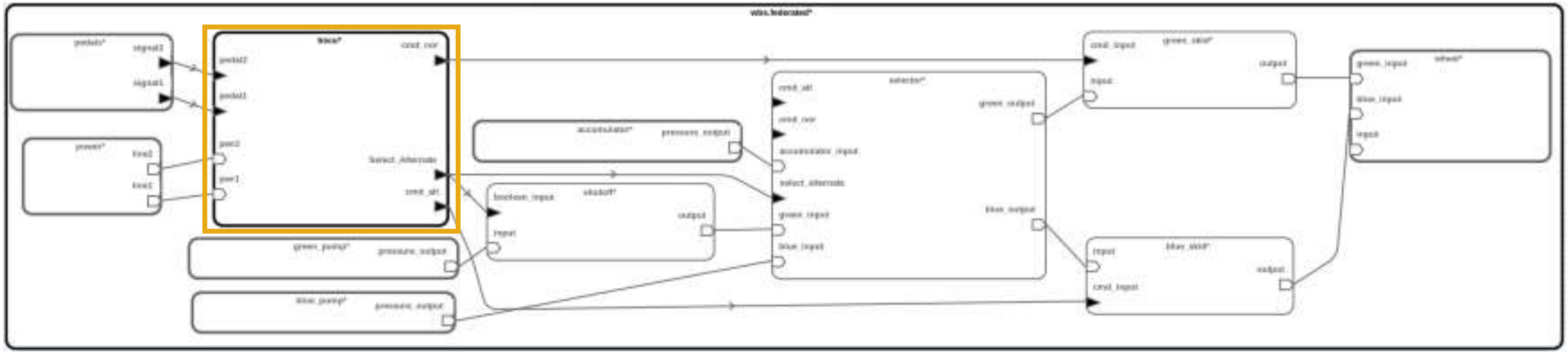
[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

The wheel brake system

© 2019 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release
and unlimited distribution.

The wheel brake system



Two subsystems (command and monitor) + common platform

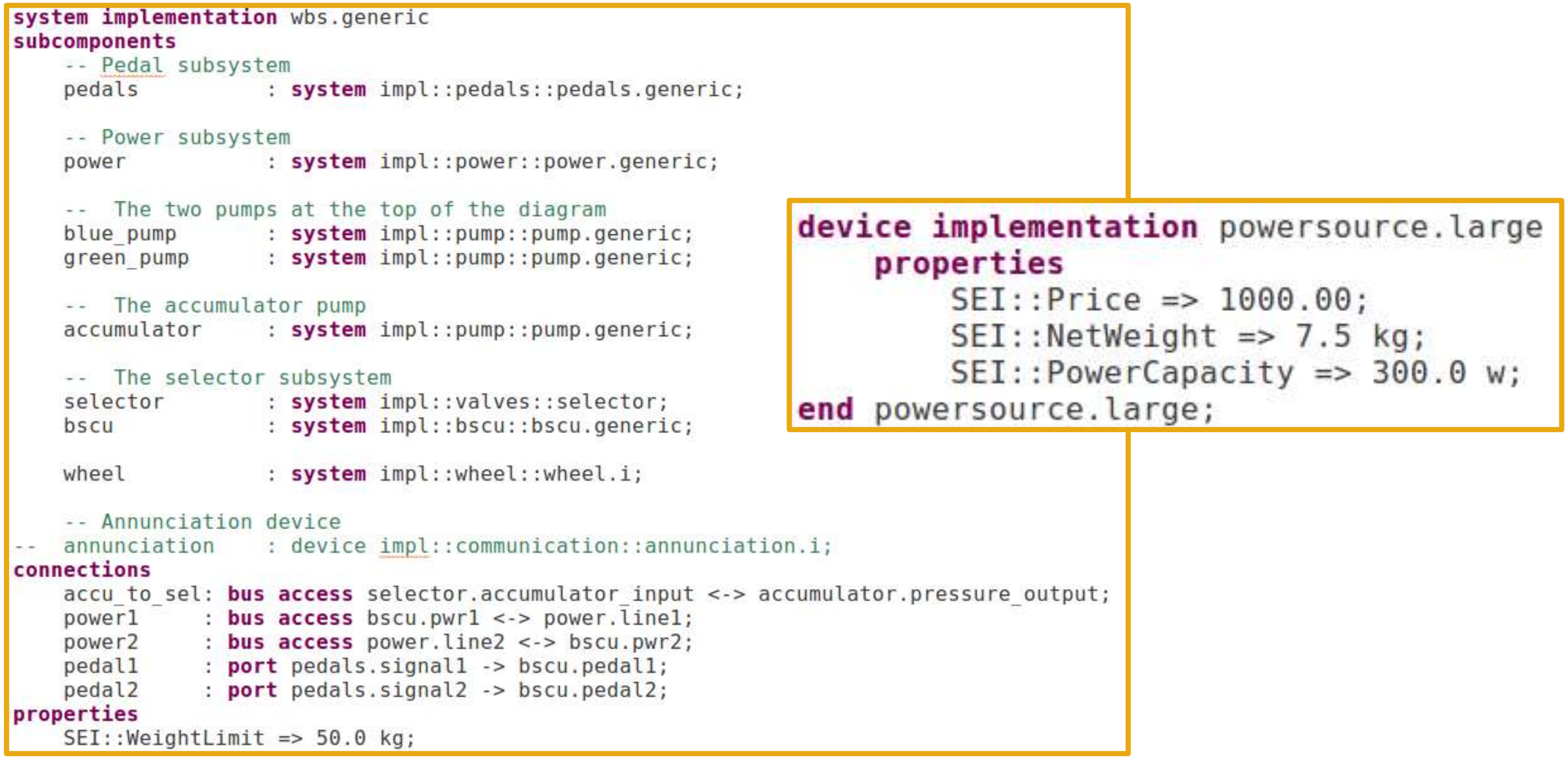
- Two monitor implementations, two command implementations
- Platform varies in power budget, wiring gauge, CPU architecture
  - Multiple CPUs must have the same architecture
  - Power required by CPUs must match platform provisions

… and that's just one component!

© 2019 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Architecture Analysis and Description Language



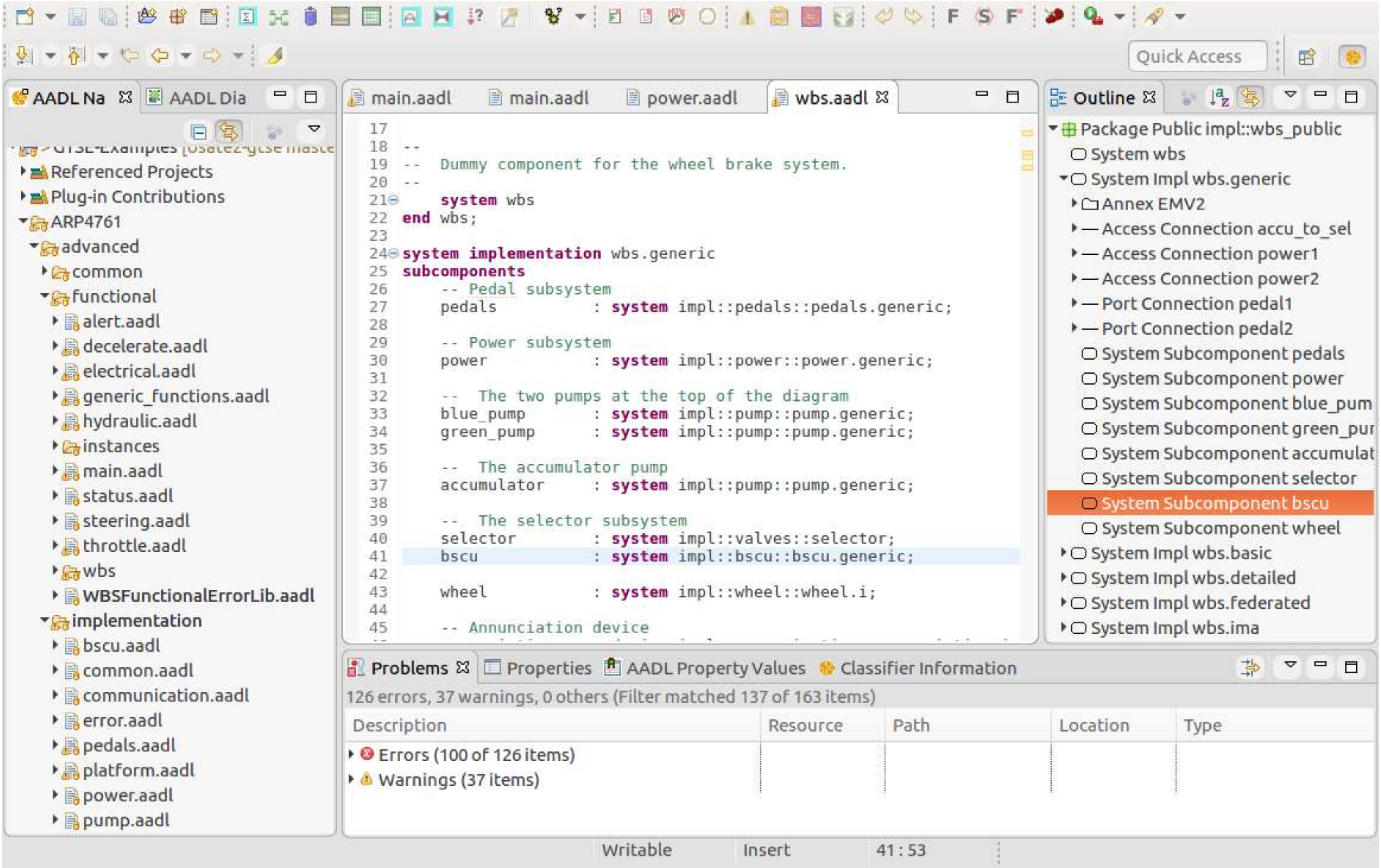
```
system implementation wbs.generic
subcomponents
    -- Pedal subsystem
    pedals              : system impl::pedals::pedals.generic;

    -- Power subsystem
    power               : system impl::power::power.generic;

    --  The two pumps at the top of the diagram
    blue_pump           : system impl::pump::pump.generic;
    green_pump          : system impl::pump::pump.generic;

    --  The accumulator pump
    accumulator         : system impl::pump::pump.generic;

    --  The selector subsystem
    selector            : system impl::valves::selector;
    bscu                : system impl::bscu::bscu.generic;

    wheel               : system impl::wheel::wheel.i;

    -- Annunciation device
--  annunciation        : device impl::communication::annunciation.i;
connections
    accu_to_sel: bus access selector.accumulator_input <-> accumulator.pressure_output;
    power1     : bus access bscu.pwr1 <-> power.line1;
    power2     : bus access power.line2 <-> bscu.pwr2;
    pedal1     : port pedals.signal1 -> bscu.pedal1;
    pedal2     : port pedals.signal2 -> bscu.pedal2;
properties
    SEI::WeightLimit => 50.0 kg;
```

```
device implementation powersource.large
      properties
            SEI::Price => 1000.00;
            SEI::NetWeight => 7.5 kg;
            SEI::PowerCapacity => 300.0 w;
end powersource.large;
```

International standard (SAE AS5506C)

Used in academia, industry, government in the US, EU, China

Open Source Architecture Tool Environment



- GTSE-Examples [osate2-gtse maste
- Referenced Projects
- Plug-in Contributions
- ARP4761
  - advanced
    - common
    - functional
      - alert.aadl
      - decelerate.aadl
      - electrical.aadl
      - generic_functions.aadl
      - hydraulic.aadl
      - instances
      - main.aadl
      - status.aadl
      - steering.aadl
      - throttle.aadl
      - wbs
      - WBSFunctionalErrorLib.aadl
    - implementation
      - bscu.aadl
      - common.aadl
      - communication.aadl
      - error.aadl
      - pedals.aadl
      - platform.aadl
      - power.aadl
      - pump.aadl

```

--
-- Dummy component for the wheel brake system.
--
	system wbs
end wbs;

system implementation wbs.generic
subcomponents
	-- Pedal subsystem
	pedals          : system impl::pedals::pedals.generic;

	-- Power subsystem
	power           : system impl::power::power.generic;

	--  The two pumps at the top of the diagram
	blue_pump       : system impl::pump::pump.generic;
	green_pump      : system impl::pump::pump.generic;

	--  The accumulator pump
	accumulator     : system impl::pump::pump.generic;

	--  The selector subsystem
	selector        : system impl::valves::selector;
	bscu            : system impl::bscu::bscu.generic;

	wheel           : system impl::wheel::wheel.i;

	-- Annunciation device
```

- Package Public impl::wbs_public
  - System wbs
  - System Impl wbs.generic
    - Annex EMV2
    - Access Connection accu_to_sel
    - Access Connection power1
    - Access Connection power2
    - Port Connection pedal1
    - Port Connection pedal2
    - System Subcomponent pedals
    - System Subcomponent power
    - System Subcomponent blue_pum
    - System Subcomponent green_pur
    - System Subcomponent accumulat
    - System Subcomponent selector
    - System Subcomponent bscu
    - System Subcomponent wheel
  - System Impl wbs.basic
  - System Impl wbs.detailed
  - System Impl wbs.federated
  - System Impl wbs.ima

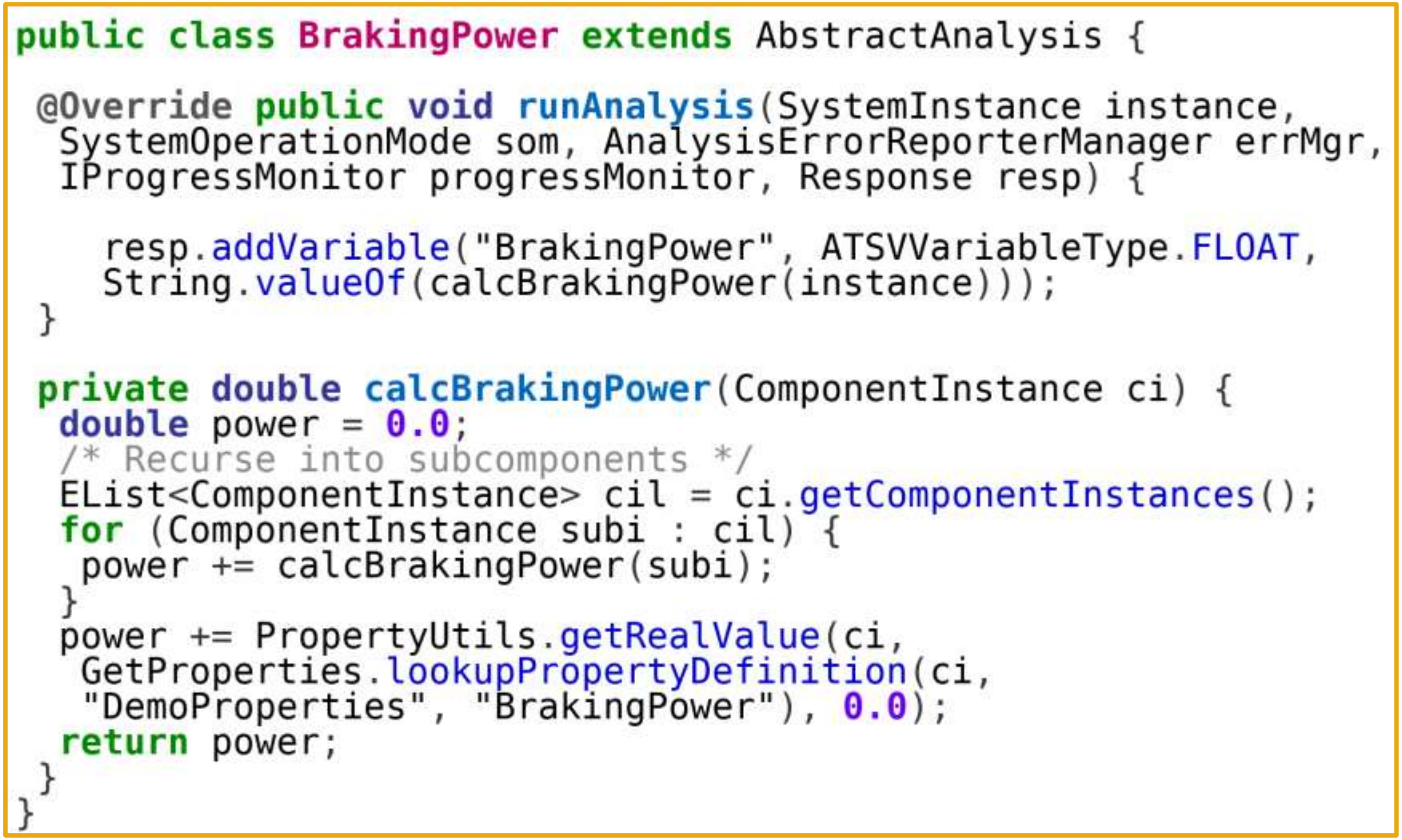
126 errors, 37 warnings, 0 others (Filter matched 137 of 163 items)

Description	Resource	Path	Location	Type
Errors (100 of 126 items)				
Warnings (37 items)				

© 2019 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Example Domain-Specific Plugin



```
public class BrakingPower extends AbstractAnalysis {

 @Override public void runAnalysis(SystemInstance instance,
  SystemOperationMode som, AnalysisErrorReporterManager errMgr,
  IProgressMonitor progressMonitor, Response resp) {

    resp.addVariable("BrakingPower", ATSVVariableType.FLOAT,
    String.valueOf(calcBrakingPower(instance)));
 }

 private double calcBrakingPower(ComponentInstance ci) {
  double power = 0.0;
  /* Recurse into subcomponents */
  EList<ComponentInstance> cil = ci.getComponentInstances();
  for (ComponentInstance subi : cil) {
   power += calcBrakingPower(subi);
  }
  power += PropertyUtils.getRealValue(ci,
   GetProperties.lookupPropertyDefinition(ci,
   "DemoProperties", "BrakingPower"), 0.0);
  return power;
 }
}
```

© 2019 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Outline

A Wheel-Braking System

Designing by Shopping

Guided Architecture Trade Space Exploration

Designing by Shopping (Balling)

What’s wrong with optimization?

- “A priori articulation of preference” (Hwang and Masud) is hard.

How do we fix it?

- Provide a range of options so users can intuitively understand tradeoffs
  - Options should be *pareto optimal*

Think of buying a shirt on Amazon…

- It’s hard to envision the perfect shirt without seeing any examples
  - And even if you do, what are the odds it exists?
- Look at some examples (yellow vs blue shirts) then refine your search
  - Repeat until you’re satisfied

Penn State’s ARL Trade Space Visualizer

Java based software for design-by-shopping.

Includes both a range of evolutionary algorithms and a variety of visualizations.

Evaluated in aeronautics and aerospace domains.

AIAA 2006-2139

Trade Space Exploration of a Wing Design Problem Using Visual Steering and Multi-Dimensional Data Visualization

Timothy W. Simpson, Daniel E. Carlsen, and Christopher D. Congdon
The Pennsylvania State University, University Park, PA 16802, USA

Gary Stump and Michael A. Yukish
The Applied Research Laboratory, State College, PA 16804, USA

I. Introduction

Visual Steering ... Space Expl... Sampling

Copyright © 2003 by ASME

A Configuration Language for AADL

An AADL Model

```
package P

    system S
    end S;

    system implementation S.i
        subcomponents
          sub: processor Intel;
    end S;

    processor Intel
    end Intel;

    processor implementation Intel.i3
    end Intel.i3;

    processor implementation Intel.i5
    end Intel.i5;

end P;
```

Assign a component implementation
and a property value

```
configuration C1 extends S.i {
    sub => Intel.i3;
    #SEI::Weight => 0.2 kg;
}
```

Extend a configuration and override an assignment
Assign a property in a nested configuration

```
configuration C2 extends S.i with C1 {
    sub => Intel.i5 {
        #SEI::MIPSCapacity => 1500 MIPS;
}   }
```

Parameterized configuration
with list of valid choices

```
configuration C3 (
    proc: processor Intel
             from (Intel.i3, Intel.i5)
) extends S.i {
    sub => proc;
    #SEI::MIPSCapacity => 1000MIPS;
}
```

© 2019 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Outline

A Wheel-Braking System

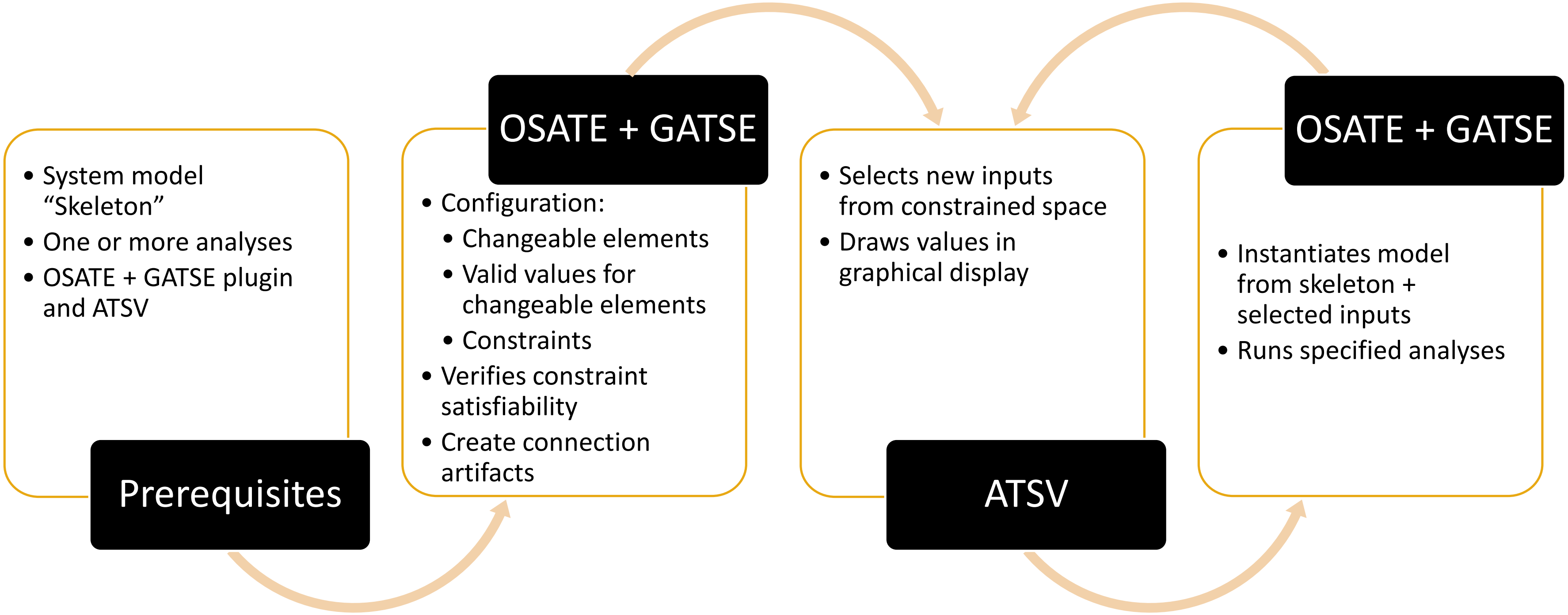
Designing by Shopping

Guided Architecture Trade Space Exploration

© 2019 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

GATSE: Workflow



Prerequisites

- System model "Skeleton"
- One or more analyses
- OSATE + GATSE plugin and ATSV

OSATE + GATSE

- Configuration:
  - Changeable elements
  - Valid values for changeable elements
  - Constraints
- Verifies constraint satisfiability
- Create connection artifacts

ATSV

- Selects new inputs from constrained space
- Draws values in graphical display

OSATE + GATSE

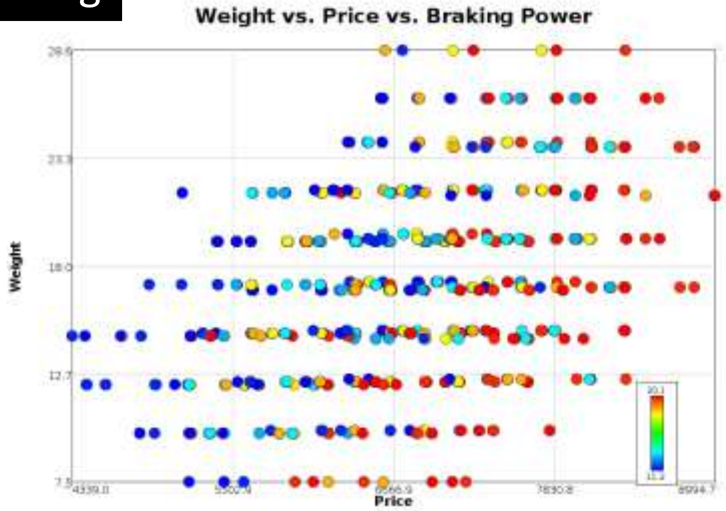
- Instantiates model from skeleton + selected inputs
- Runs specified analyses

© 2019 Carnegie Mellon University

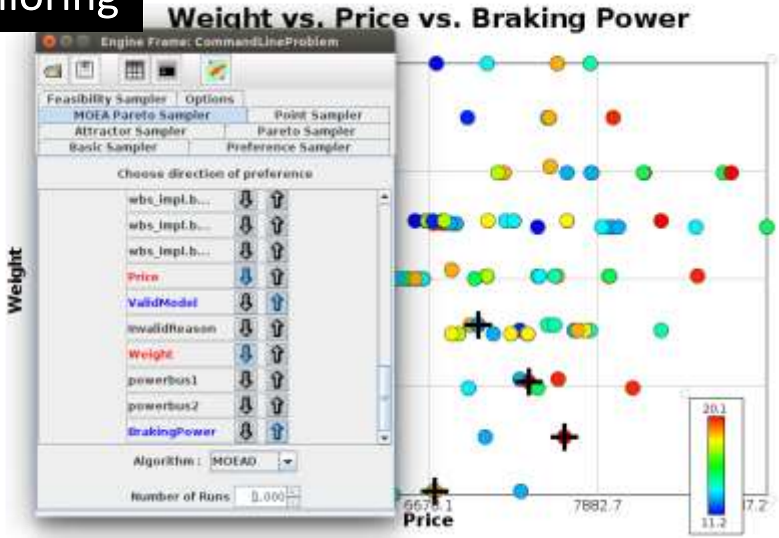
[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

GATSE (ATSV): In action

Viewing



Tailoring

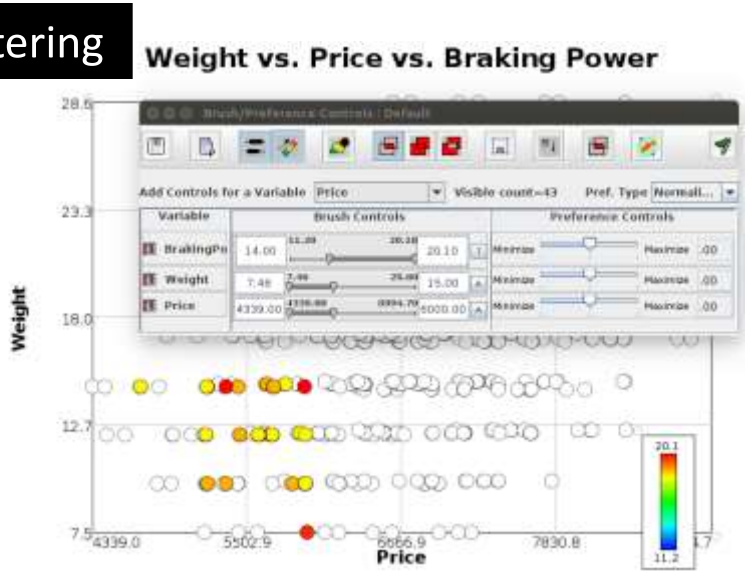


Choose direction of preference

Algorithm : MOEAD

Number of Runs 1.000

Filtering

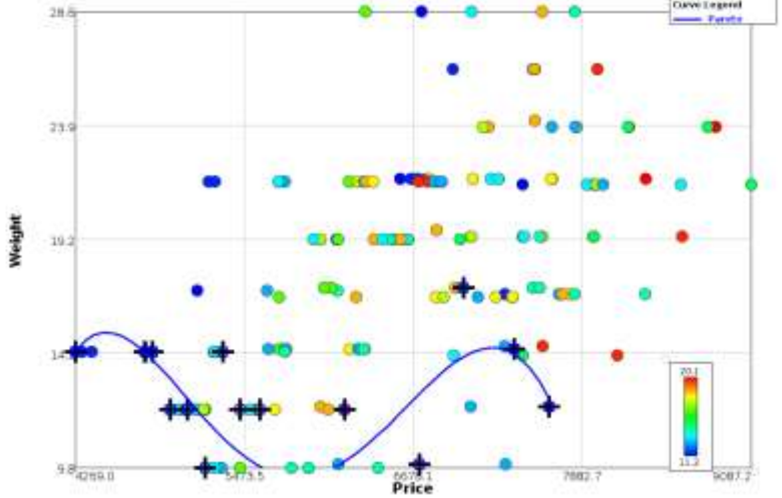


Weight vs. Price vs. Braking Power
Brush/Preference Controls : Default
Weight
Price
28.6
23.3
18.0
12.7
7.5
4339.0
5502.9
6666.9
7830.8
20.1
11.2

Add Controls for a Variable Price Visible count=43 Pref. Type Normali...

Variable	Brush Controls		Preference Controls	
BrakingPo	14.00	20.10	Minimize	Maximize .00
Weight	7.48	15.00	Minimize	Maximize .00
Price	4339.00	6000.00	Minimize	Maximize .00

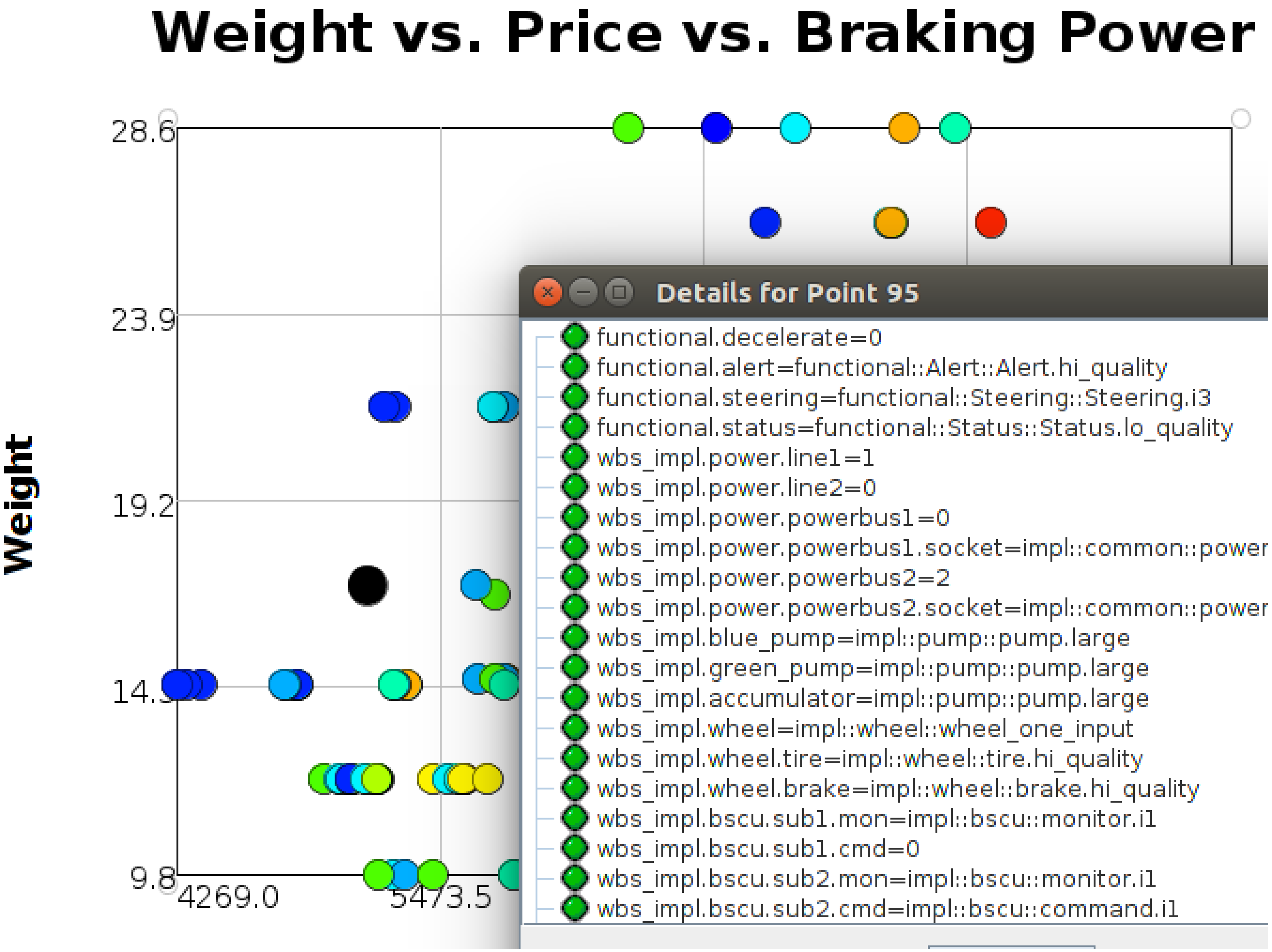
Pareto Frontier*



© 2019 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

GATSE (ATSV): In detail



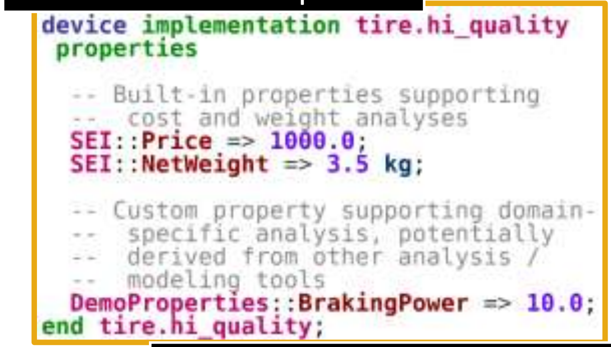
- functional.decelerate=0
- functional.alert=functional::Alert::Alert.hi_quality
- functional.steering=functional::Steering::Steering.i3
- functional.status=functional::Status::Status.lo_quality
- wbs_impl.power.line1=1
- wbs_impl.power.line2=0
- wbs_impl.power.powerbus1=0
- wbs_impl.power.powerbus1.socket=impl::common::power
- wbs_impl.power.powerbus2=2
- wbs_impl.power.powerbus2.socket=impl::common::power
- wbs_impl.blue_pump=impl::pump::pump.large
- wbs_impl.green_pump=impl::pump::pump.large
- wbs_impl.accumulator=impl::pump::pump.large
- wbs_impl.wheel=impl::wheel::wheel_one_input
- wbs_impl.wheel.tire=impl::wheel::tire.hi_quality
- wbs_impl.wheel.brake=impl::wheel::brake.hi_quality
- wbs_impl.bscu.sub1.mon=impl::bscu::monitor.i1
- wbs_impl.bscu.sub1.cmd=0
- wbs_impl.bscu.sub2.mon=impl::bscu::monitor.i1
- wbs_impl.bscu.sub2.cmd=impl::bscu::command.i1

© 2019 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

The GATSE Vision

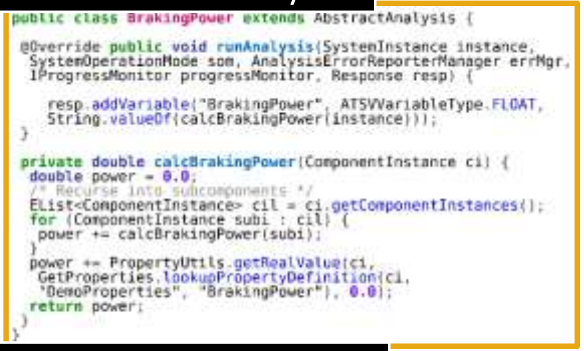
AADL: Custom Properties



```
device implementation tire.hi_quality
 properties

   -- Built-in properties supporting
   --  cost and weight analyses
   SEI::Price => 1000.0;
   SEI::NetWeight => 3.5 kg;

   -- Custom property supporting domain-
   --  specific analysis, potentially
   --  derived from other analysis /
   --  modeling tools
   DemoProperties::BrakingPower => 10.0;
end tire.hi_quality;
```

Used By

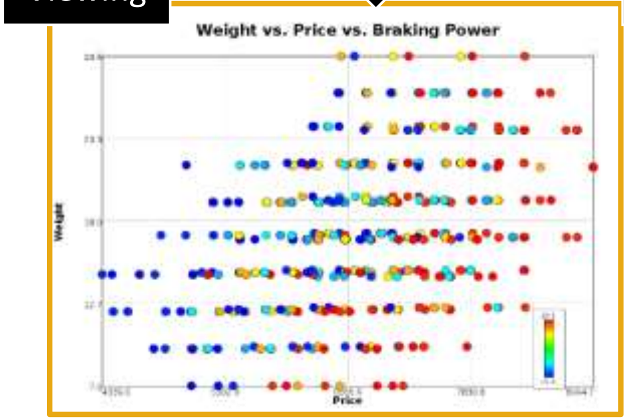
OSATE: Custom Analyses



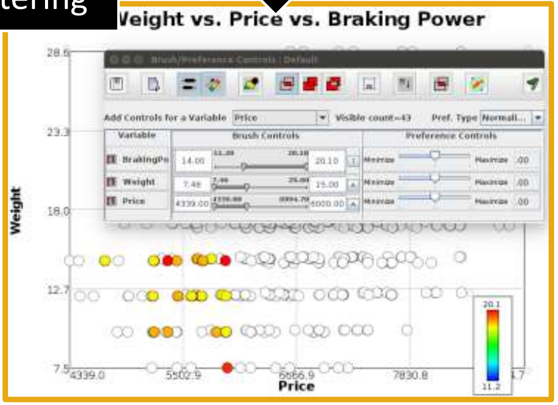
```
public class BrakingPower extends AbstractAnalysis {

 @Override public void runAnalysis(SystemInstance instance,
  SystemOperationMode som, AnalysisErrorReporterManager errMgr,
  IProgressMonitor progressMonitor, Response resp) {

    resp.addVariable("BrakingPower", ATSVVariableType.FLOAT,
    String.valueOf(calcBrakingPower(instance)));
 }

 private double calcBrakingPower(ComponentInstance ci) {
  double power = 0.0;
  /* Recurse into subcomponents */
  EList<ComponentInstance> cil = ci.getComponentInstances();
  for (ComponentInstance subi : cil) {
   power += calcBrakingPower(subi);
  }
  power += PropertyUtils.getRealValue(ci,
   GetProperties.lookupPropertyDefinition(ci,
   "DemoProperties", "BrakingPower"), 0.0);
  return power;
 }
}
```

Enables (ATSV)

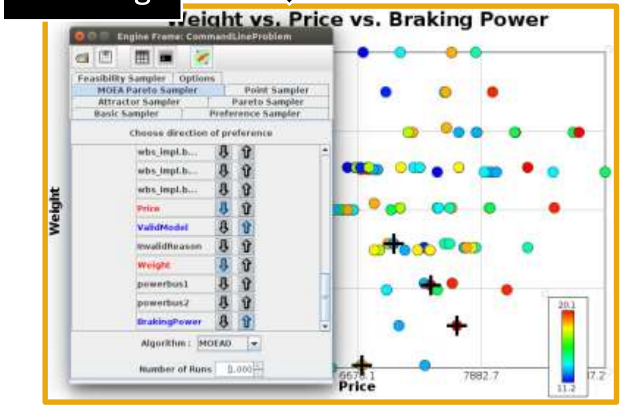
Viewing



Filtering



Tailoring



© 2019 Carnegie Mellon University

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Guided Architecture Trade Space Exploration

Fusing Model Based Engineering and Design by Shopping

Sam Procter

Lutz Wrage

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.