Experience Report for Modeling the Single and Two Door Power Window in ClaferMPS

By: Jordan A. Ross

How to Read this Document

- This is an experience report for creating a single and two door power window model in ClaferMPS.
- I (Jordan Ross) tried to be as specific as possible in my details. I apologize for any proof reading mistakes.
- I have named/color-coded some of the steps/entries as follows:
    - **Observations**: These are observations that I have made while working with the tool. Some of these observations can be classified as bugs, possible improvement, or limitations.
    - **Mitigation**: These are nested under observations which were added after the modeling was complete to provide some comments for how the bug could/was fixed, or how the tool could be improved, or how to address a limitation.
    - **Classification**: These are nested under observation which were added after the modeling was complete to provide the classification of the observation:
        - Bug – An unintentional defect in the tool that was not planned and can readily be fixed.
        - Requested Feature/Possible Improvement – A feature that the user requests or a possible improvement that could be made to aid the user in using the tool.
        - Insight – An insight into using the tool or a feature that stood out to the user as *nice to have*. It also includes an insight for ways to model with using the tool.
        - Limitation – A limitation of the tool in which no mitigation strategy has been identified.
        - Limitation (solved) – A limitation of the tool in which a mitigation strategy has been identified which solves the limitation or provides an equivalent alternative with no side effects.
        - False observation – This is an observation that was made but ended up being solved. The mitigation strategy for these observations details how it could have been avoided.
        - Possible Pitfall – An observation which could lead to problems in the future by either allowing incorrect models or the user not knowing something is wrong in the model.
    - **Bug**: This was a type of observation that was classified as a software bug in the tool which was usually minor. These were either annotated as fixed or not.
    - **Fixed**: Notation used to note that the bug was fixed. These were not added in chronological order.
    - **Tip**: A general tip for other Clafer MPS users that I found useful.
- Some bugs are repeated in the document to show when they were fixed.

Input Material

The material that the power window model was based off of was the plain Clafer encoding of the power window system detailed in a technical report GSDLAB-TR-2016-05.

Planned Steps for Modeling Power Window in ClaferMPS

1. Start with the single door power window and model the concrete architecture.
2. Take the concrete single door power window, generalize it, and instantiate it for the single driver door.
3. Extend the generalized architecture for the front passenger.
4. Generate the Clafer and try running some of the queries.

Actual Steps for Modeling Power Window in ClaferMPS

1. I started by following the GitHub instructions and getting the ClaferMPS project built and ready for import.
2. I created a new Project called "CaseStudies" that would house both case studies.
    a. I added the following as project libraries:
        i. org.clafer.mps (path to the "org.clafer.mps" directory inside the ClaferMPS directory)
        ii. org.mbeddr (from the mbeddr documentation)
        iii. org.mbeddr.plugins (from the mbeddr documentation)
3. I created a new Solution called "PowerWindow" which would contain our power window models.
    a. Under the power window solution, I added the following as dependencies:
        i. org.clafer
        ii. org.clafer.core
        iii. org.clafer.expr
        iv. org.clafer.architecture
4. I then created a model under the PowerWindow solution named "DriverDoor" to represent the concrete driver door power window model.
    a. Under the model properties I added the following *used languages*:
        i. org.clafer
        ii. org.clafer.core
        iii. org.clafer.expr
        iv. org.clafer.architecture
5. I next created a "ClaferModule" under the driver window for the feature model named DriverWinSysFM.
    a. I pulled up the menu and select "Feature Model" for the first line
        i. **Observation:** I noticed that there were a lot of possibilities I could have chosen for my element.
            1. **Mitigation**: You could have more defined clafer modules (i.e one for feature model, power topology) have some wizard for setting it up. Also in the dialog screen details could be discussed about what the module is.

2. **Classification**: Requested Feature/Possible Improvement

b. I then proceeded to add the features. I started with the basic and express down features.
   i. **Observation:** I noticed that the text "Feature" does not show up when I create a feature.
      1. **Mitigation**: I think that this is a preference of the user. It would be good to be able hide or show "Feature" based on some intention menu setting for a feature model.
      2. **Classification**: Requested Feature/Possible Improvement
c. I then created the express up feature nested under the express down.
d. I then went back and clicked on the features and the "light bulb" then selected to make the express down and express up optional.

6. I then rebuilt the PowerWindow solution and previewed the generated clafer:
   a. I got an error: [ArchitectureUtil.getModuleDependencyByName] Could not find org.clafer.architecture.baseConcepts
   b. The generated Clafer made DriverWinSysFM an abstract instead of a concrete Clafer…
   c. Also the generated Clafer had not abstract clafers for feature and feature model
7. I then created another Clafer module under DriverDoor named DriverWinSysFAA for the functional analysis architecture
   a. I pulled up the menu and selected FA Architecture
   b. I then created the Switch, Arbiter, WinControl, Motor, and Current Sensor
   c. I then created a nested functional analysis architecture for the pinch detection
      i. I then selected it to make it optional
   d. I then created the connectors between the functions, nesting the pinch detection ones inside the architecture.
      i. **Observation:** I can't use any comments inside the functional analysis architecture to comment on the partition of the architecture and make notes.
         1. **Mitigation:** Use the mbeddr documentation engine. Add margin comments. This is actually quite nice because you can attach them to specific model elements.
         2. **Classification:** Limitation (solved)
   e. I then used the EAST-ADL project to ensure that I had all the connections defined between the components.
   f. I then build the solution again and previewed the generated Clafer.
      i. I got the same error as before and the generated Clafer also did not have any meta-model elements but had the constraints for sender and receiver oddly enough.
8. I then wanted to link the feature model to the functional analysis architecture.
   a. I added DriverWinSysFM to the import list of DriverWinSysFAA
   b. Then wrote the constraint using the intention and autocomplete menus.
   c. **Observation:** This was actually quite nice because I didn't have to look back at the nesting hierarchy in the feature model.
      i. **Mitigation:** None.
      ii. **Classification:** Insight

d. I however got an error saying that I could not express equivalence. This is a bug in the tool.

9. I then created another Clafer module under DriverDoor named DriverWinSysHA for the hardware architecture
    a. I pulled up the menu and selected HA architecture
    b. I then created the device node classification name DriverWinSysDN
        i. **Bug:** Device node topology should be renamed device node classification
            1. **Fixed**
    c. I then created the Switch device node
        i. I was unable to make the switch either electric or smart
        ii. **Observation:** I found it counter intuitive how to add multiple device node type alternatives.
            1. **Mitigation:** Edit the intention menu to have a two-step process of "add device node type" > "<select device node type>". This could be done multiple times to give multiple alternatives.
            2. **Classification:** Insight
    d. Bug: Wire connector should not be present in the possible elements in a device node classification.
        i. **Fixed**
    e. I then created the Motor device node.
    f. I then created the DoorInline device node.
        i. **Observation**: When the intention menu comes up for the device node type I should be able to select from the list. Not just have it be smart because then I have to add a second type (power) then delete the first one (smart). This is not easy or intuitive at all.
            1. **Mitigation:** This is identical to 9.c.ii
            2. **Classification:** This is identical to 9.c.ii
    g. I then created the DoorModule device node
        i. I used the intention menu to make it optional as well.
    h. Next I want to include the BCM and EC.
        i. **Observation:** I would like to distinguish between a local and a central node (an outside node that I am using), however I don't see any way to do this.
            1. **Mitigation:** This could have been avoided by knowing that references exist and better documentation. An example of this should be highlighted in our documentation.
            2. **Classification:** False observation
        ii. I left them out for now.

10. I then created a new model, Environment to model the components that were not owned by the power window but other systems inside the power window solution.
    a. Inside the Envrionment model I created a ClaferModule Car. Inside the Car I defined two top level device nodes BCM and Electric Center.

11. Next, I asked Eldar about the abstract clafers and I realized that I had been forgetting to model the most top level element "System". It is in the system that I can generate the correct Clafer

with the system being top level concrete. Additionally, this made me realize that I needed to change my "Environment".

12. I then backtracked to the DriverDoor model and created another ClaferModule named DriverWinSys.
    a. Inside the module I instantiated a System named DriverWinSys
        i. Inside the system I first created a feature model name DWinSysFM which inherited from DriverWinSysFM (which I had to the import list first)
        ii. Next I rebuilt and tried generating Clafer to see what I got. Besides the error that I am getting for not finding baseConcepts the generated Clafer looks good.
        iii. Next I instantiated an architecture, DWinSysArch.
        iv. Inside the architecture I instantiated the functional analysis architecture DWinSysFA of type DriverWinSysFAA.
        v. I did the same thing for the hardware architecture.
13. Next, I wanted to just express the deployment of functions to the device node topology. I created a new ClaferModule DriverWinSysDpl which deploys the DriverWinSysFAA to the DriverWinSysHA.
    a. I also added DriverWinSysFAA and DriverWinSysHA to the import list.
    b. Next I created the simple deployment constraints, such as the WinSwitch is deployed to the Switch, WinMotor is deployed to the Motor, and the CurrentSensor is deployed to where the WinMotor is deployed.
        i. I was unable to do this however because there was no child "deployedTo" or anything in the autocomplete for WinSwitch. This is probably due to the BaseConcepts error that I have.
14. Bug Fix: I figured out that I need to include org.clafer.architecture.baseConcepts as a dependency to my "PowerWindow" solution.
    a. When I rebuild and generate Clafer for the system I get the correct types.
15. Now returning to the deployment, I still don't see "deployedTo" as an option for intentions.
    a. After talking to Eldar I realized that I was supposed to specify the deployment in the functional analysis architecture.
    b. I then deleted the DriverWinSysDpl Clafer module and started adding the deployments to the functional analysis architecture.
        i. I also included DriverWinSysHA as an import to the DriverWinSysFAA
        ii. **Observation:** I can't express that "CurrentSensor is deployed to the device node that WinMotor is deployed"
            1. **Mitigation:** It is possible to express but you have to use a Clafer constraints. This might cause problems later on because users are expressing constraints in two separate places.
            2. **Classification:** Possible Pitfall & Limitation
        iii. **Observation:** I think it would be good to add "deployed To" in the intention menu.
            1. **Mitigation:** Add a deployment option in the intention menu of analysis functions, functional devices, and function connectors.
            2. **Classification:** Feature Request/Possible Improvement

c. For the analysis functions, I would like to deploy to the BCM. For now, what I am going to do is just deploy them to the Car BCM instead of a reference owned by the driver device node topology.
    i. I had to first add Environment as a dependency to the "DriverDoor" model.
    ii. Then I could add "Car" as an import to DriverWinSysFAA
d. I allow the analysis functions to be deployed to any of the 4 device nodes (Switch, Motor, DoorModule, BCM)

16. I then rebuilt the functional analysis architecture and only generated Clafer for the FAA. I got an error saying that it failed to generate the text for the DriverWinSysFAA.cfr and DriverWinSys.cfr.
    a. **Bug:** The error is due to the generators not working correctly with multiple MPS models. Therefore, what I have to do to get around it right now is to move the ClaferModule "Car" inside the DriverDoor model and then use "virtual packages" to separate the Car from the other modules.
17. I then performed the workaround in 16.a and I was able to build and generate successfully.
    a. **Observation:** Doing this refactoring was quite tedious as I had to delete all instances of EC and BCM from the model then re add them in. At least, MPS caught the fact that the old instances of BCM were referring to incorrect elements.
        i. **Mitigation:** No real workaround for this. The fact is that MPS handles this better than most tools. Maybe have some migration scripts for tasks like this if they become abundant.
        ii. **Classification:** Limitation
18. After previewing the generated text, I realized that the BCM and EC were abstract. This stems from the assumption that all top level model elements turn into abstract Clafers except for system. Therefore, I have to define a complete system and the required layers to model just the BCM and EC for the environment.
    a. **Observation:** I am not sure how I feel about this "Environment". I like to think of the environment model as a "test stub". So I would like a quick and dirty way to instantiate my outside components that may change at a later time.
        i. **Mitigation:** This is a limitation in comparison to Clafer because in Clafer I don't have to follow the hierarchy. We could change the reference model to allow this but after some discussion we will leave for now.
        ii. **Classification:** Limitation
    b. **Observation:** When I moved the BCM I had to change the references to it all over again. I understand why but I rather just have a single place where I have to change, like I can do when I have reference for central components in plain Clafer.
        i. **Mitigation:** See 9.h.i.1
        ii. **Classification:** False observation.
19. At this point I wanted to see if I could generate instances. So I generated the Clafer for DriverWinSys and copied to sublime text and tried to compile.
    a. **Bug:** In the generated text:
    abstract DataConnector : HardwareConnector
        [some fc, : FunctionConnector | fc.deployedTo = this]
    Should be:
    abstract DataConnector : HardwareConnector

[some fc : FunctionConnector | fc.deployedTo = this]

i. **Fixed**

b. **Bug:** The generated Clafer should use * not "-1" in upper bound cardinalities.

i. **Fixed**

c. Also how are we going to handle the [this.parent] ?

At this point I took a look at the reference model used in the DSL and realized how different it was from the final reference model that was used in SoSYM paper.

Things that need to be changed in the meta-model:

- Remove implementation layer and all references to it
- We need to add the logical bus bridge element or we can't fully model the case study for the power window.
- Remove all domain specific constructs such as ECU, Switch, Motor. Those should be in a separate module that could be imported by a user if they wish

20. So since the Clafer generators are not completely finished/correct I am going to continue with modeling w/o checking my solution for generated instances.
    a. **Observation:** This is a something that I did a lot in Clafer: write a small model then test. I am not seeing that workflow here (mostly because it is not supported). Also I am thinking that generating instances will be useless feedback because when a user only has the FAA and device nodes done, the lack of a communication topology will throw off the generated instances without the use even knowing it I think.
        i. **Mitigation:** This is a big user request that would require extensive work to implement. No workaround for the time being.
        ii. **Classification:** Feature Request/Possible Improvement & Limitation
21. Up to this point I have the device node classification modeled but no other elements. Therefore, next I am going to model the communication topology.
    a. I start with adding a communication topology, DriverWinSysCT, inside the hardware architecture.
    b. Then I add the bus connector for the door name localBusConnector.
        i. **Bug:** "Realized by" should not be in the bus connector definition, just the connects part.
            1. **Fixed**
        ii. **Bug:** I can't model the constraint that our local bus connector is either LIN or low speed CAN
    c. Next, I added the discrete data connectors between each of the nodes, except the DoorModule and BCM because they will always communicate via the bus.
        i. **Bug:** "Discrete Wire Connector" should be renamed to "Discrete Data Connector"
            1. **Fixed**
        ii. **Bug:** There is no endpoints or sender and receiver for the discrete data connector but it does exist in the meta model….
            1. **Fixed**

d. **Observation:** After I modeled these, I wanted to switch to the graphical perspective to check and see if I had all the connectors modeled.
    1. **Mitigation:** Needs to be implemented.
    2. **Classification:** Bug
    ii. **Bug:** This is not implemented so I could not comment on the benefit.
e. **Observation**/**Bug:** The Discrete Wire Connector and all other data connectors (besides the bus should say “Discrete Wire Connector <name> connects <device node A> and <device node B>”
    i. **Mitigation:** Nothing. Bug was fixed.
    ii. **Classification:** Bug

22. Next I moved on to the power topology since all of the kinks were worked out (besides the graphical projection not being implemented).
    a. I first modeled the device node power connectors for each of the devices.
        i. I first made the connectors then selected which ones where optional (which was all of them).
        ii. Next I added the constraint that if the door module was present then I must have the door module device power connector.
        iii. Next I added the constraint that I had the motor and switch device powers if they were smart.
            1. **Bug:** Why is “type” in the constraints a keyword?
                a. **Fixed**
        iv. Next I also added the constraint that if I had any of the device power connectors leaving the door inline then I needed the device power going to the inline from the EC.
        v. **Observation:** I had this constraint here and not in the deployment because these constraints do not depend on the functional analysis architecture in any way unlike the load power.
            1. **Mitigation:** None.
            2. **Classification:** Insight
    b. Next I modeled the load power connectors for the different configurations.
        i. **Observation:** If I try to define all the MPS connectors with variable connector ends it won’t work because it requires us give concrete device nodes. We could remedy this by allowing for a set to be given to the endpoints of the connector where only one is chosen when instantiated.
            1. **Mitigation:** By choosing this way of modeling it allows for the user to express all of the variability. However, it does require additional components that modeling with plain Clafer does not. A possible improvement would be to allow more than on device node in the endpoints.
            2. **Classification:** Limitation & Insight
        ii. Another thing I thought of doing is to elicit the different topologies using a nested power topology that has the XOR group cardinality. This is a much more verbose technique but this is what I went with for now until the previous point is addressed.

iii. **Observation:** Now that I have it implemented, this would be a case where I would want to test out if my logic is sound (i.e. if it makes sense what I have and if I get instances). What would be very cool in future work would be for me to highlight the section of the topology that I want to test and it would export to Clafer (or run in suite) with the necessary "test stubs" such that I can test out the logic. I think this would be an invaluable tool for a user coming up with/debugging a correct model
    1. **Mitigation:** This is a huge user request, would require extensive work and design.
    2. **Classification:** Feature Request/Possible Improvement
iv. I did "test" it out by building the model and viewing the generated Clafer for the hardware architecture. Since I know what I am looking for in terms of plain Clafer I can debug/analyze the generated model to see if what I had was correct. Which I deemed was in fact correct.
    1. **Observation:** Something that I noticed when looking at the generated Clafer though that worried me is the notion of "namespace" and how it is handled in MPS. I am concerned that duplicate names could arise in a generated model.
        a. **Mitigation:** One way to do this is to allow for fully qualified path names (or show them in the autocomplete).
        b. **Classification:** Limitation
v. Since I am leaving out the constraints (based on the deployment of the WinControl) to the deployment layer I am done with the power topology.
vi. **Observation:** I would like to visualize this graphically. Something that again would be nice is for me to highlight a portion of a model and just project the selected text to the graphical view.
    1. **Mitigation:** Have some option for viewing selected elements in the architecture DSL graphical projection.
    2. **Classification:** Feature Request/Possible Improvement

23. At this point I have the device node classification, the power topology, and the communication topology done so I am done with the hardware architecture. Therefore, the next thing I need to do is to add the deployment layer.
24. What I forgot though was that the deployment layer doesn't exist since all deployment of analysis functions, functional devices, and function connectors is captured at the FAA. The one thing that is not captured is the relationship between the WinControl deployment and the power topology.
    a. Also I have not specified the deployment for the function connectors so I did that first. I tried to be as restrictive as possible for the deployments (i.e. I just didn't give all possible communication connectors to the function connectors which I could have since the solver would have picked appropriately but it would have been slower).
    b. In addition, while I was adding the deployment connectors I wanted to encode the constraint that all analysis functions have to be deployed to smart devices only.
        i. **Observation:** Since a user can't edit the meta-model in any way I can't write a constraint nested under analysis function that it must be deployed to a smart

device. Therefore, for each analysis function modeled I have to write the constraint manually. This is very tedious and could cause errors over time.

1. **Mitigation:** Allow users a way of writing constraints in some separate module (like the QA or preferences module) using the reference model elements. These could be written in the System module with explicit for all quantification.
2. **Classification:** Limitation

c. Next I needed to add the constraints based on the deployment of the WinControl to the power topology configurations. I decided that it was best to house the constraint in a deployment element so that it is in one place. Therefore, I created a new Clafer module DriverWinSysDpl.
   i. Nested inside I wrote the constraints for the power topology configuration based on the win control deployment.
   ii. **Bug:** When I tried to model the constraints it did not work due to a type error.
      1. **Fixed**
   iii. With the bug fixed I was able to finish the model.

d. Lastly, I added the deployment to the system.
   i. **Bug:** How do I add a generalized deployment? I could not add a super node to my deployment.
      1. **Fixed**

25. To this point I have the entire structure of the driver power window system modeled. Now, the next task is to add the quality attributes to the model.
26. The first quality attribute I decided to add was mass of the system. Under the driver door system, I created a new quality attribute model named DriverQualityAttributes.
    a. I am not sure how to go about this so I had to refer to the example that Eldar had.
    b. I first clicked "Visible". Next I added quality attributes for the device nodes.
       i. When I first tried to create the quality attribute I was getting errors. I had to add "org.clafer.architecture" as a Dependency (not only a language) in the PowerWindow solution.
       ii. **Observation:** This step was a little buggy
          1. **Mitigation:** User testing.
          2. **Classification:** Bug
       iii. I also added the cost and warranty cost quality attributes for device node when I was creating them
    c. Next I did the same thing for hardware connector (which bus connector and discrete data connector inherit from).
       i. I also included cost and length.
       ii. **Observation:** It would be really good to have the UML model showing the inheritance hierarchy and the defined concepts made available to the users so they have some reference to work with.
          1. **Mitigation:** Have the UML or Clafer model embedded somehow such that it could be referenced easily. One thing we discussed is to hyperlink from a syntax element like "Analysis Function" and it would show the

details of what an analysis function is and how it fits into the reference model.
2. **Classification:** Feature Request/Possible Improvement

d. Now that I had the quality attributes defined I also needed to define some constant values for the mass per unit length. I did this by adding a Preferences module named "DriverQualityPreferences" that I could use to define the quality attribute constants.
   i. Again I had no idea what I was doing here so I had to look at Eldar's example.
   ii. I first created a ConstantGroup named "MassPerUnitLength"
   iii. Next I created a Constant LoadPowerConnector (and tabbed it once so it was nested under the constant group.
   iv. I did this for the remaining constants under mass per unit length

e. Now that I had the constants defined, I needed to add the mass for each of the device nodes.
   i. I did this using the intention menu for the device node which was quite nice.
   ii. I also did this for the BCM and EC nested under the environment.
      1. **Observation:** I am wondering if we will have problems with quality attributes when we are sharing the "Car" with multiple systems.
         a. **Mitigation:** Unsure. There will somehow have to only be one Car clafer module defined between two or more models. Maybe move the Car clafer module into its own model.
         b. **Classification:** Possible Pitfall

f. Next I need to add the mass for the bus connector. I did this by first modeling the length for bus connector.
   i. **Observation:** I then realized that I had not given any "types" to the bus connector. Eldar should implement this like the device node.
      1. **Mitigation:** None.
      2. **Classification:** Bug.
   ii. I can't even handle it in constraints because "type" does not show up under bus connector.
      1. **Bug:** I should be able to access "type" under BusConnector using constraints.
         a. **Fixed**
   iii. **Observation:** The other thing is that the mass of the bus is dependent on the type. I would be great if I could write the mass equations inside the meta-model elements like I do in Clafer. Otherwise it will get really messy and cumbersome because I will have to write multiple constraints for each bus.
      1. **Mitigation:** Allow users to write constraints in the QA module pertaining to reference model elements that hold for every instance of the reference model element.
      2. **Classification:** Limitation
   iv. I decided to leave the bus for now and moved on to the discrete data connectors.

g. I first modeled the lengths for the discrete data connectors.

h. Next I modeled the mass of the discrete data connectors using an equation based on the discrete data connector mass per unit length connector for *each* of the discrete data connectors.
    i. **Observation:** Again, if we could write this equation once it would be much more user friendly. If I can do it in Clafer, it would be nice to be able to do it here.
        1. **Mitigation:** Identical to 26.f.iii
        2. **Classification:**
    ii. **Observation:** In the autocomplete when choosing "length" for the equation, it is unclear what element the "length" belongs to.
        1. **Mitigation:** Allow the use of fully qualified paths.
        2. **Classification:** Possible Pitfall

i. I then moved on to the power topology. The device power is quite straight forward (and is just like the discrete data connector.
    i. **Observation:** I was just thinking that it would be good to model/show the unit system for the quality attributes such that if one engineer comes up with the quality attributes, the person who enters the values knows what units the values are in.
        1. **Mitigation:** Include this unit system in the QA table.
        2. **Classification:** Feature Request/Possible Improvement
    ii. **Observation:** A very cool feature would be allowing users to enter the values in floating point numbers then when the generate Clafer they would select some precision and the generate Clafer would use integers and scale them appropriately.
        1. **Mitigation:** A feature that would require a lot of design and implementation.
        2. **Classification:** Feature Request/Possible Improvement

j. The load power was also fairly straightforward due to how we modeled (i.e. the concrete connectors were modeled in each of the possible topologies.

k. After adding in these quality attributes I realized how large the hardware architecture was getting. Therefore, I decided to split it into separate modules. I first did this by creating a new ClaferModule for device node classification
    i. **Observation:** This was very messy at first… But I used "Fix Clafer reference" and it worked really well actually.
        1. **Mitigation:** No mitigation strategy. Identical to the issue in 17.a
        2. **Classification:** Identical to 17.a
    ii. **Bug:** I actually backed this out because when I tried to do it for communication topology there was a problem because it was not nested under a hardware architecture.
        1. **Fixed**

l. I then rebuilt the solution and realized that there was an error in the mass equation referring the length. So I removed the mass for now.

27. I then decided to add the cost quality attribute. I first created the new constants (since I had the quality attributes defined).
    a. **Bug:** I was having trouble removing the indent but I eventually got it.

b. I then added the cost to the device nodes but not to any other component due to the bug in the equations for mass.
c. Adding the cost for the Switch, DoorModule, DoorInline, BCM, and EC is straight forward.
d. For the Motor however we use an "if statement" base of if the device is smart or not.
e. I then build the project again and observe the generated Clafer for correctness.

28. Next, I do the exact same thing for warranty cost for the device nodes.
    a. I model the replacement cost and ppm but I can't write the equations for warranty cost because of the error with using equations that I had for mass.
29. To this point I have all of the quality attributes modeled except for latency.
30. For latency I need to add latency quality attributes to analysis function, functional devices, function connectors, and data connectors.
    a. I start with data connectors where I add transferTimePerSize to DataConnectors
    b. I then add the latency parameters for functional devices and analysis functions.
        i. **Bug:** Because of how the domain model is defined, the quality attributes defined for the analysis functions are inherited by the functional devices which we don't want because baseLatency should not exist in functional device.
            1. **Fixed**
    c. Lastly, I add the speed factor parameter for the device node.
31. Next I enter the concrete values for the latency parameters.
    a. **Observation:** Again here I want to write some general constraints, such as if a device node is not smart then the speed factor should be zero among others.
        i. **Mitigation:** Identical to 26.f.iii
        ii. **Classification:** Identical to 26.f.iii
    b. **Bug:** When I am writing my latency equation for analysis functions I don't have access to the deployedTo's quality attribute speedFactor.
        i. **Fixed**
    c. **Bug:** I also couldn't enter the transfer time for the bus connector due to the variation in the bus type.
        i. **Fixed**
    d. **Bug:** I also couldn't write the latency equations for function connectors.
        i. **Fixed**
    e. **Bug:** When you type "this.deployedTo.type" under a function connectors you get access to "smart, elecr, power" which is incorrect.
        i. **Fixed**
32. So at this point I have the quality attributes somewhat modeled but I am missing the equations due to bugs. The last thing I need to do is to come up with "totalCarCost", "totalCarMass", and the latency equations.
    a. I did this by adding a cost, mass, and warranty cost quality attributes to the system using the DriverQualityAttributes module.
    b. **Bug:** I am not sure how to write the constraints to sum all of the costs of the Driver device nodes since we are using nested abstracts.
33. After discussing with Michal and Eldar, references do exist so I am going to implement a local reference in DriverWinSysDN to the BCM and EC.

a. **Observation:** Update the intention menu for a reference to say “Add target for reference” or “Add device node target”
    i. **Mitigation:** None.
    ii. **Classification:** Bug
        1. **Fixed**
b. **Observation:** One of the techniques I came up with for modeling architectures is that no quality attributes should be associated to the references. Can we enforce this to help users model better?
    i. **Mitigation:** Disallow quality attributes being applied to references.
    ii. **Classification:** Limitation

34. At this point the complete driver door has been modeled with all of its layers. The next step is to generalize the architecture so we can extend it for two power window examples.
35. To do this I am going to create a new model inside the power window named “TwoDoor”.
    a. I add the same languages as I did in step 4.a
36. Next I want to copy the Car Clafer Module and its virtual package into my new model.
    a. I chose not to include the import DriverDoor model
    b. **Observation:** When I just copy and paste into my new model I get errors that cost and mass are out of scope. This has to be something wrong with the quality attributes model.
        i. When I just re-write it seems to work just fine so something is staying attached when it shouldn’t…
        ii. **Mitigation:** None
        iii. **Classification:** Bug
    c. By re-writing the BCM and ElectricCenter I have the virtual package and Car module moved into my new project.
37. Next I want to copy the DriverWinSysFM to the TwoDoor model and rename it for our generalized architecture.
    a. I don’t have any issues or errors because there are no quality attributes for the features.
    b. Next I use the rename option and rename DriverWinSysFM (the feature model element) to WinSysFM.
    c. I use the same option to rename the module name to WinSysFM.
38. Next I want to package my generalized modules together so I right clock on WinSysFM and “Set Virtual Package” and name it to “GeneralizedArchitecture”.
39. I then copy and paste DriverWinSysFAA to the new virtual package and rename. However, I also want the values of the qualities to get copied so what I decided to do is first copy the quality attributes model (and preferences) into the GeneralizedArchitecture virtual package.
    a. So after the QA and preferences modules have been imported, I copy and paste DriverWinSysFAA.
    b. The only errors I have are the import references are incorrect. Thus, I remove the old ones and add in the WinSysFM. I still need the hardware architecture before I can import it.
    c. **Observation:** By adding the QA first, I was able to not have any errors w.r.t. the qualities and all of the values got copied correctly!
        i. **Mitigation:** None.

ii. **Classification:** Insight

d. Also, there was not error for the Car import so I used "CTRL + Click" to navigate to the module to ensure it was, in fact, the one for the two door model which it was.
e. I then used the renaming function on the module name and FAA name.
f. At this point all of the deployment references are incorrect, which is expected and I leave it for now.
g. However, there is an error with the feature model constraint (again expected so I update this).
    i. I am thinking this could have been avoided though if I would have not renamed DriverWinSysFM until after I had the FAA copied.

40. Next I copy the hardware architecture DriverWinSysHA to the generalized architecture package.
    a. The only error that I get is for the reference to electric center. The reason I don't get one for BCM is the name is the same as the reference thus we don't get an error.
        i. **Observation:** This could be a major pitfall to users potentially.
            1. **Mitigation:** Use fully/partially qualified path names.
            2. **Classification:** Possible Pitfall
    b. Before I update the reference I rename the module and hardware architecture
    c. Then I go back to the FAA and import the correct module.
    d. I then use the rename functionality on the driverWinSysDN and it suggests a refactoring for all the device nodes used in the file which I accept by clicking "do refactor".
    e. I then apply renaming to the communication topology.
    f. I decide to wait on the power topology until I copy and paste the deployment since I know it uses the naming.
    g. I update the references targets manually for the BCM and EC.
41. I then copy and paste the deployment DriverWinSysDpl to the generalized package.
    a. I update the imports first.
    b. I then have to rename the FAA and HA used in the deployment and rename the module and deployment.
    c. I then have errors in the constraints.
    d. I return to the WinSysHA and rename the power topology in which it does not catch any refactoring's.
    e. I then go back to the deployment and have to manually apply the renaming to the constraint elements.
        i. **Observation:** It would be good to have some mechanism that resolved references for us.
            1. **Mitigation:** Implement some migration scripts to cover typical migration tasks such as this.
            2. **Classification:** Feature Request/Possible Improvement
42. I then return to the FAA and have to update all of the references manually after including WinSysHA as an import.
    a. **Tip:** Use "CNTRL + SPACE" to find the names quickly when the old reference is (partially) selected.
43. Lastly, I rename DriverQualityAttributes to GeneralQualityAttributes and do the suggested refactoing. I do the same thing for the preferences.

44. I then rebuild the solution to verify that everything is working right now.
    a. Besides the errors I was getting before the solution is good.
45. Next, I need to create the two concrete systems for the driver and passenger.
    a. I first do the driver by copy and pasting it to the two door model (outside of the virtual packages.
    b. I remove all the old imports and replace them with the new ones.
    c. I then renamed all of the generalized architectures and deployments.
        i. Due to a previous bug 24.d.i I was unable to use the generalize deployment so just ignored deployment for now.
    d. Since the driver is just the generalized architecture there is nothing I need to do.
46. I then created a new Clafer module for the front passenger door.
    a. I first just instantiate the generalized layers of the architecture.
    b. Next since the passenger should not have more features than the driver I add constraints as content to the FPWinSysFM.
        i. Before I do I need to include DriverWinSys as an import since I need to reference its feature model.
        ii. **Bug:** I can't write the constraint because I don't have access to nested features under DriverWinSysFM for some reason. I am assuming it has to do with inheritance.
            1. **Fixed.**
    c. Moving on, I needed to add a functional device to represent the functional device of the driver side window switch and the additional function connector.
        i. I added this as content inside the FAA
            1. **Observation:** The WinArbiter used as the receiver for the function connector might not be the correct namespace.
                a. **Mitigation:** Use fully/partially qualified names.
                b. **Classification:** Possible Pitfall
    d. **How do I refine the device node classification**?
        i. Not sure…
47. At this point I have waited for Eldar to fix the bugs that I have pointed out to him. (Reference email). Now that the bugs are worked out I need to update my models to conform to the changes he has implemented first.
    a. **Observation**: Making changes to the reference meta model is disastrous from a user standpoint when you have to update.
        i. **Mitigation:** Identical to 17.a
        ii. **Classification:** Identical to 17.a
    b. I had to pretty much remake the driver and two door system.
    c. I started with the driver door module. The FM was okay so I moved to the FAA and had to delete and rewrite the whole model. There was some bug which wouldn't let me delete an element without deleting the entire model.
        i. **Bug**: Can't specify that the current sensor should be deployed to what WinMotor is deployed to.
            1. **Fixed**
            2. Maybe I can if I use the deployment rules?

a. You can. This seems disjoint.
b. **Observation**: This could be dangerous. You are not writing deployments in two separate places and you could easily write conflicting constraints.
    i. **Mitigation:** Similar to 15.b.ii
    ii. **Classification:** Possible Pitfall
c. **Observation**: I used MPS built in documentation to write a comment that the deployment rule was specified somewhere else.
    i. **Mitigation:** None.
    ii. **Classification:** Insight

ii. **Observation**: Do we even need to specify the deployment of function connectors. If don't want to restrict the *valid* topologies only the valid ones can be synthesized due to the constraints.
    1. I left out the deployment of function connectors this time around.
    2. **Mitigation:** None
    3. **Classification:** Insight

d. I then moved the hardware architecture and the Device node classification was fine. I however did need to rewrite the power and communication topologies.
    i. **Bug**: Making a new line after a list of connections for bus connector
        1. **Fixed**
    ii. **Bug**: Should HardwareConnector, HardwareDataConnector show up?
        1. **Fixed**
    iii. **Bug**: Does Bus connector not inherit from hardware connector?
        1. **Fixed**
    iv. **Observation**: The hints (i.e. the placeholder) don't help users what valid values are okay. What does help though is the autocomplete.
        1. **Mitigation:** Improve the names
        2. **Classification:** Feature Request/Possible Improvement
e. I then moved to the deployment and removed the old references.
f. At this point the driver door module was back to a stable state. I still had not implemented anything new or fixed any of the missing things from previous steps. Next I moved on to fixing the two door model.
g. The feature model again was good but the FAA was broken.
    i. I fixed this by just copy and pasting from the single door case.
h. I then moved to the hardware architecture where I just copied and pasted again.
i. I then updated the deployment in the same way.
j. Lastly I updated the Driver and FrontPassenger models.
k. Everything to this point looked good so I rebuilt and get generator errors but Eldar told me that those would be there for now.

48. Now that I have the model back to a working state I began trying to finish implementing the two door system before talking the quality attribute updates. Going back to the front passenger system module I was able to write the constraints I needed.

a. **Observation**: I was unsure if I needed "this" in my constraints to get the correct "expressDown".
    i. **Mitigation:** Not sure that this is a valid observation because this is more of a Clafer thing.
    ii. **Classification:** False Observation
b. I then moved to the FAA and added the functional device and connector. However, it is still unclear which "WinArbiter" dWinReq is referring to.
    i. **Observation**: Two things would help this:
        1. Be able to write FPWinSysFAA.WinArbiter
        2. Visualizing in the diagrams.
        3. **Mitigation:** Implement this.
        4. **Classification:** Feature Request/Possible Improvement
    ii. I took the second option and attempted to visualize the two diagrams. What I found was that it was incorrect
        1. **Bug**: When you look at the architecture DSL perspective you don't see WinArbiter come up inside a box. And again, as a user/modeled I am confused to what the DriverWinSwitch is connected to.
            a. **Fixed**.

49. Next I moved on to the hardware architecture. I referred back to step 46.d where I wasn't sure how to refine the device node classification. It turns out its impossible with how we have it setup. Instead, I switched to how I modeled in Clafer where I generalized each of the hardware architecture components separately (device node classification, power topology, and communication topology).
    a. I did this by creating separate Clafer modules for the generic components and just copied and pasted what I had in the HA. This was a trivial and easy to do task.
    b. Since the hardware architecture is no longer common (different doors have different components) we remove it.
        i. **Observation**: By removing the hardware architecture it broke our generalized deployment since we no longer have a target for the deployment. This is bad because now we have to have users put the power topology constraints in two places. We could get around this by placing these constraints in other modules.
            1. This is definitely a place for improvement and a concrete use case where MPS actually hinders us in using a common core architecture.
            2. **Classification:** False Observation.
        ii. Based on my observation I decided to move the specific deployment constraints that were in WinSysDpl to each of the systems deployment (i.e. DWinSysDpl and FPWinSysDpl).
        iii. I also defined the specific device node classification, comm topology, and power topologies for both systems.
    c. At this point I have the common architecture instantiated for both with extensions only applied to the functional analysis architecture. Now I need to add the additional hardware architecture components needed for the front passenger.
    d. For the FPWinSysDN I needed to define a reference to the drivers switch and use it for the deployment of the DriverWinSys functional device.

i. **Bug**: I am unable to set the target of this reference to DriverWinSys.DWinSysHA.DWinSysDn… in the syntax.
    1. To get around this I add the constraint to the deployment.
ii. **Bug**: Why can you add a bus connector to a deployment?
    1. **Fixed**

e. For the FPWinSysCT I needed to define addition discrete wires along with the bus bridge.
    i. I also added the lengths to the discrete wires.
    ii. **Bug**: I was unable to add the bus bridge as it was not implemented yet.
        1. **Fixed.**

f. I didn't have to add any extra deployment constraints because the solver should synthesize correctly based on the underlying reference model constraints.

g. **Observation**: I am not sure if I made this observation already, but I think we need a defined domain model (i.e. class diagram) that shows the relationship between the elements. This is because I am bias (I developed the ref model in Clafer thus I know how it works) but a new user will have no idea what is what.
    i. **Mitigation:** Identical to 26.c.ii
    ii. **Classification:** Identical to 26.c.ii

50. At this point I have almost all of the structure modeled for both systems. The only thing missing is the bus bridge. Now I want to update the quality attributes. To begin, when we generalized we just kept the attributes in the generalized architecture. This is not always correct because it assumes the two systems are geometrically symmetric (which they aren't) for length.
    a. We revisit the length of wires by first realizing the connections between the local nodes have the same lengths for both doors thus we can leave them. However, the distance from the inlines to the BCM and EC is different for both doors.
    b. To begin we start with the inlineEC device power connector. The length is connectors is different for the two doors.
        i. To do this we remove the quality attribute from the generalized architecture.
        ii. Next we have to just use Clafer constraints to set the value of the length to the value.
        iii. **Observation**: This seems limited and like a hack. Is there a cleaner way to do this?
            1. **Mitigation:** Unsure…
            2. **Classification:** Limitation
        iv. **Observation**: Are these constraints going to restrict variability?
            1. **Mitigation:** Possible bug?
            2. **Classification:** Bug
    c. I then continue with this for the rest of the power topology wires that need it. And then I move the communication topology and do it there as well.
    d. At this point I have the lengths correctly modeled for the two systems.
51. The next thing I want to do is revisit the bugs (not related to generators) that I came across in early steps. For each bug I comment on the status and any changes I made to the model (unless previously covered.)
    a. **Bug:** Device node topology should be renamed device node classification

i. **Fixed**

b. Bug: Wire connector should not be present in the possible elements in a device node classification.
   i. **Fixed**

c. **Bug:** I can't model the constraint that our local bus connector is either LIN or low speed CAN
   i. **Fixed**
   ii. Updated both the driver and two door models.

d. **Bug:** The Discrete Wire Connector and all other data connectors (besides the bus should say "Discrete Wire Connector <name> connects <device node A> and <device node B>"
   i. **Fixed**

e. **Bug:** Why is "type" in the constraints a key word?
   i. **Fixed**

f. **Bug:** How do I add a generalized deployment? I could not add a super node to my deployment.
   i. **Fixed**
   ii. I did not end up using this however, see step 49.b

g. **Bug:** I should be able to access "type" under BusConnector using constraints.
   i. **Fixed**

h. **Bug:** I actually backed this out because when I tried to do it for communication topology there was a problem because it was not nested under a hardware architecture.
   i. **Fixed.**

i. **Bug:** Because of how the domain model is defined, the quality attributes defined for the analysis functions are inherited by the functional devices which we don't want because baseLatency should not exist in functional device.
   i. **Fixed.**

j. **Bug:** When I am writing my latency equation for analysis functions I don't have access to the deployedTo's quality attribute speedFactor.
   i. **Fixed.**
   ii. I added the equations to the analysis functions for both models.

k. **Bug:** I also couldn't write the latency equations for function connectors.
   i. **Partially Fixed.**
   ii. There is an issue where the function connectors are only being deployed to bus connectors or something. The relationships is not correct.
   iii. **Bug**: There is a bug with the function connector deployed to reference target.

l. **Bug:** When you type "this.deployedTo.type" under a function connectors you get access to "smart, elecr, power" which is incorrect.
   i. **Fixed.**

m. **Bug:** I am not sure how to write the constraints to sum all of the costs of the Driver device nodes since we are using nested abstracts.

52. Since I have verified what bugs are fixed next I wanted to try and calculate the total device node mass, cost, and warranty cost. I did this by adding the quality attributes in the table for the device node classification element.
    a. I also had to update the device nodes such that I modeled the warrant

b. **Observation**: The warranty cost is a prime example of a constraint that I want to write once and have it apply to each device node. Other examples of this are connector mass and cost.
    i. **Mitigation:** Identical to 26.f.iii
    ii. **Classification:** Identical to 26.f.iii
c. I then had to manually add up the individual local components. I did this at system definition of both systems.
d. **Observation**: Eliciting all the device nodes nested under the device node classification is tedious and I should be able to write a single constraint (similar to that in 52.b) where I say sum(DeviceNode.mass).
    i. **Mitigation:** Identical to 26.f.iii
    ii. **Classification:** Identical to 26.f.iii
e. At this point I had the totals for the device node classification. I copied and pasted this work to the single door model as well.

53. Next I moved down to the power topology and did the same thing for modeling the total cost and mass. Warranty cost is not considered at the power topology.
    a. Same observation. Mass and Cost equations are so tedious to write.
    b. What I did for the load power topology is since totalCost and totalMass was defined for power topology I just added those to each of the xor group load power topologies and to the load power topology itself.
    c. **Observation**: This is a pretty neat technique of modeling because you get fidelity of constraining each of the power topologies (i.e. you could choose that the load power topology must cost less than X).
        i. **Mitigation:** None
        ii. **Classification:** Insight
    d. I did not do this for the single door yet because this is a pretty tedious task.
54. Next I moved on to finishing the quality attributes for the communication topology. I first started with mass and cost.
    a. I did the same thing I did for the power topology and defined the equations for the discrete data connectors.
    b. **Observation**: For the bus connector I had to use implies (based on the type of bus used for the mass and cost equations. This is way too tedious…
        i. Luckily since there are just two alternatives I can just use an “if statement”. Not sure what I would do if there as more than two possible types for the bus.
        ii. **Mitigation:** You can always just use plain Clafer constraints but that would be very messy in my opinion
        iii. **Classification:** Limitation
    c. I then totaled up the values.
    d. Lastly I needed to add the transferPerSize quality attribute values to the connectors. I had to set all the discrete data connectors to zero. I had the same issue in 54.b where I need to assign different transferPerSize to different types of buses.
    e. At this point I have the hardware architecture components done for quality attributes.

55. Next I added total mass, cost, and warranty cost to the hardware architecture and then to systems. At this point I have the totals modeled completely for the two door case. I need to still update the single door model.
56. The last thing I need to do to the model is to model the latency of the system. The last elements that need to have the correct latency equations are the function connectors. I did this for each of the function connectors.
    a. See bug in 51.k.ii
    b. Now that each of the necessary functional analysis architecture elements have latency information we need to come up with the timing chains.
    c. To do this is a little weird. How I have to do this is by defining quality attributes for each of the chains.
    d. I don't model the requirements because I will just leave that for the queries.
    e. **Observation**: Do I need constraint (implies) for the position latencies?
        i. **Mitigation:** Unsure…
        ii. **Classification:** Possible Pitfall
    f. **Bug**: Max and Min of a set have not been implemented in the MPS language.
    g. **Note**: This is a note to myself that I think I need to play around with defining a reference model element for a timing chain in plain Clafer and investigating if it's possible to make the modeling nicer for a user.
57. At this point I have all the qualities and structure modeled for the two door model. Thus all I am waiting on is for any bus that remain. I will add a next step with the bugs that are fixed when they are.
58. Eldar has implemented a few more bug fixes and the logical bus bridge. I had to update the project which again was a disaster.
    a. **Bug**: There is a bug with the function connector deployed to reference target.
        i. **Fixed**
    b. **Bug**: Why can you add a bus connector to a deployment?
        i. **Fixed**
    c. **Bug**: Should HardwareConnector, HardwareDataConnector show up?
        i. **Fixed**
    d. **Bug**: Does Bus connector not inherit from hardware connector?
        i. **Fixed**
    e. The above bug fixes required making changes to the connectors.
    f. Once the model was back into a stable state. I moved on to adding in the logical bus bridge for the two door model.
        i. **Observation**/**Bug**: Again we have a problem where we can't specify the context of what "locaLowSpeedBus" we want to connect.
            1. **Mitigation:** Identical to 46.c.i.1
            2. **Classification:** Identical to 46.c.i.1
            3. **Fixed.**
        ii. I also added in the gatewayTransferTimePerSize quality attribute and the equation for the correct modeling.
        iii. **Observation**/**Bug**: The constraints are not enough nested under the logical bus bridge to support not having endpoints defined for the bridge.

1. **Mitigation:** Either find a way to encode the constraint in the reference model or add in endpoints to the syntax.
2. **Classification:** Bug

iv. **Bug**: I don't have access to this.bus under logical bus bridge so I can't express my equation.
1. **Fixed**

v. **Bug**: Load power connector should not show up in communication topology.
1. **Fixed**

59. At this point the structure has been modeled completely. The only thing that remains is bug fixes for quality attributes and the generators.
60. After some discussion with Eldar and Michal, I realized that observation 49.b.i was actually false. Thus, what I did was I created a new Clafer module under generalized architecture called WinSysHA, as well as one for deployment. The hardware architecture then had the three components defined using the generalized versions (WinSysDN, WinSysPT, and WinSysCT).
    a. Then inside the generalized WinSysDpl I moved the common constraints out of the individual systems that were the same.
    b. Then inside the individual systems I made WinSysDpl a super node of both deployments.
    c. Then inside the individual systems I made WinSysHA a super node of both hardware architectures.
61. Also I realized that for the locallowspeed door bus I didn't have to use Clafer constraints but could use the intention menu to add the types of bus.
62. Eldar made another release of claferMPS in which I updated so that I could access the new diagrams.
63. First thing I did after updating is to update my model which was not too bad since no changes to the reference model where made.
    a. Bug: The migration still was not perfect even when not changing the meta-model as I had to remake a device power connector and fix the broken references to it.
64. Next since Eldar has included the diagrams for the hardware architecture I revisited the early observations regarding the hardware architecture. In addition I also included all of the diagrams (at their respective stages) to the experience report. Note these were added out of chronological order.
    a. Revisiting observation 22.b.vi:
        i. **Bug:** The scaling of the names in the device nodes is off and very small.
        ii. **Bug:** There is a bug where the labels are not being placed correctly.
        iii. **Observation:** I did not find it beneficial to the have device node classification in the diagram. I thought I was a waste of space.
            1. **Mitigation:** Remove device node classification.
            2. **Classification:** Insight
        iv. **Observation:** I like the nesting but I also think it would be beneficial to have an option to not show nesting and instead double click a fragment to open its "contents" like OSATE does in their view.
            1. **Mitigation:** Implement.
            2. **Classification:** Requested Feature/Possible Improvement

v. I also want to re-inforce 22.b.vi where we could select a portion of the module to view not the entire module.

65. Eldar made changes again but this time to the generators along with a few other small things. I proceeded by revisiting the bugs presented earlier and marking them fixed or not.
66. One of the changes that Eldar made was to allow the deployment of a functional analysis component to the deployed device of another functional analysis component. This was a completed request due to an earlier observation made.
67. After these updates I was able to build the door locks solution with zero errors. At this point we have generated Clafer but now we need to transition to using the Clafer compiler and chocosolver to validate that the generated Clafer is correct.
68. **Observation:** When we generate Clafer we only do it for 1 system at a time. Thus we have a two door system we have to create a new module which instantiates the two systems.
    a. I have not tried this yet.
    b. **Mitigation:**
    c. **Classification:**
69. Eldar has made another "release" of ClaferMPS so I updated my local install and the case studies. I then revisited the list of bugs to mark any fixed ones a fixed.
    a. One of the bugs that was fixed was being able to say "#(deployedFrom)" so that I could accurately model the data connectors.
70. At this point I am able to generate plain Clafer, and Eldar implemented a feature such that I could generate Clafer without the qualities. This is great so I could first test that I am getting the correct instances and then test with the qualities.
    a. **Bug**: The "generate quality attributes" does not consider any quality attributes set using mixed-in Clafer constraints.
        i. **Fixed**
    b. **Bug**: Bug with "smart" inconsistent definition in reference model.
        i. **Fixed**
    c. **Bug**: We are making use of redefinition which is not supported by chocosolver for the BCM and EC references.
    d. **Bug**: We have a namespace issue for the WinSysHA when we have the following names:
       abstract WinSysHA : HardwareArchitecture
       WinSysDN : WinSysDN
       WinSysPT : WinSysPT
       WinSysCT : WinSysCT
    e. I then played around with the generated Clafer and came across a couple of issues.
        i. I removed the deployment and hardware architecture from the model for debugging purposes as I thought they were causing an unsat core.
        ii. **Observation:** With those removed and continually trimming down the model I came across two mistakes that I made. The first was that none of the communication connectors were optional thus causing an unsat core. Additionally, when I was going through the plain Clafer line by line I noticed that I did not specify a deployment for the pinch detection analysis function or the position sensor. I also realized I was not including the door module in my

possible deployments. I then updated these in the MPS model and re-generated the Clafer.

1. **Mitigation:** Using the micro-level patterns (testing a small fragments at a time) I would have caught this faster I think. Static analysis could also be used to identify possible missing deployments.
2. **Classification:** Insight

iii. With these *mistakes* of mine fixed I regenerated the Clafer to observe if the deployment and hardware architecture would cause issues.

iv. The hardware architecture definition doesn't work when we don't properly nest the abstract Clafers. We need to adjust the model such that WinSysDN is nested under the hardware architecture WinSysHA and is an *extension point* of Device Node Classification.

v. I am just going to delete the hardware architecture from the generate Clafer for now. By deleting this we have a problem with the deployment. Furthermore, since we don't support reference refinement in the chocosolver yet, we can't have deployment and the generated Clafer needs to be updated to actually refine the references once implemented in chocosolver. Therefore I am just going to delete the deployment for now and move the power topology constraints to WinSysPT.

f. **Bug:** There is a bug in the generated constraints for the reference model. The following is an invalid statement.

```
abstract AnalysisFunction : FunctionalAnalysisComponent
  [FunctionalAnalysisComponent.deployedTo.type in SmartDeviceNode]
```

And should be replaced with:

```
abstract AnalysisFunction : FunctionalAnalysisComponent
  [deployedTo.type in SmartDeviceNode]
```

i. Fixed

g. **Bug:** The function connector constraint "[(sender.deployedTo , receiver.deployedTo) in (deployedTo.endpoint)]" should be nested under deployedTo.

i. Fixed

h. After manually making this changes I am able to generate instances. Now the test is to make sure I get the correct amount which I don't. I only get 80 instances.

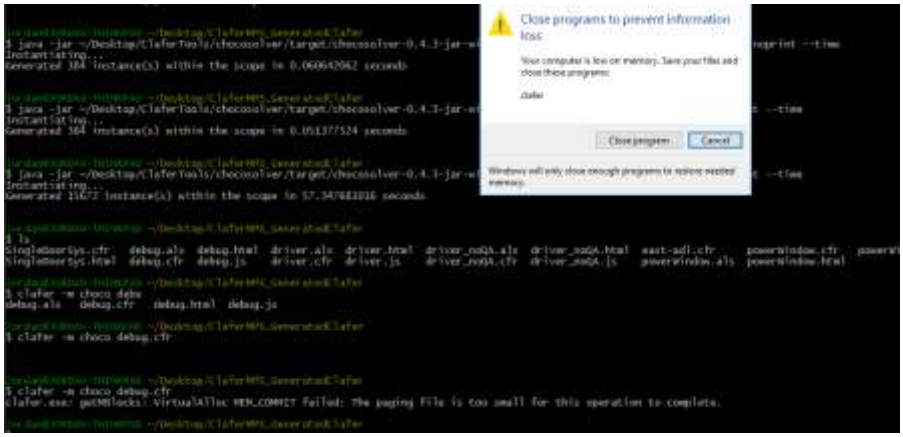
i. One difference is I have the BCM as optional in the plain Clafer case studies. I make this adjustment in the MPS model.

1. That only gets me to 160 instances.

ii. After spending time debugging I found the following issues:

1. **Bug:** Missing dref when using references in the reference model. You need it in the following constraints:
   a. [(sender.deployedTo.dref, receiver.deployedTo.dref) in (deployedTo.endpoint.dref)]
   b. [(sender.deployedTo.dref = receiver.deployedTo.dref) <=> no this.deployedTo]
   c. Fixed

2. **Bug:** Missing dref when using the BCM as a target for deployments and endpoints.
    a. **Fixed**
3. Observation: I made a *mistake* and had motorInlineDP not as optional.
4. **Bug:** I had the bus marked as LIN/LowSpeedCAN but those constraints were not generated in the Clafer.
    a. **Fixed**
5. **Bug:** The constraint "[(sender.deployedTo = receiver.deployedTo) <=> this.deployedTo]" should be replaced with: "[(sender.deployedTo.dref = receiver.deployedTo.dref) <=> no this.deployedTo]"
6. At this point I got the same number of instances (for the non-quality attribute) generated model as I did for the plain Clafer implementation.

i. Next I wanted to test the quality attribute generated Clafer and see if I got the same number of instances.
    i. I first noticed that I was missing the quality attributes for the car. Thus I update the MPS model to include the attributes.
    ii. Next, I removed the timing chains from the generated Clafer as they were going to cause issues and I was not happy with how they were modeled to begin with. I will address these later.
    iii. **Bug:** The generated if statements are not correct. Sometimes in the generated Clafer the clafers are missing.
        1. **Fixed**
    iv. I realized I made a mistake in the quality attributes and I had not included values speed factor for device nodes. I fixed this right away.
    v. After applying the refactoring's (I thought) to the model based on the non-QA model I found that the compiler was crashing.
        1. The issue was because I was dereferencing the BCM but had yet to change the "BCM->BCM" reference.
        2. 

    vi. I realized that I made a *mistake* and had not included the latency for the functional devices. Additionally, I had a baseLatency quality attribute defined for the functional device which was wrong. I fixed both right away.
    vii. I realized I made a mistake in not dividing the value of the connectors by 1000 (due to the max integer size of choco).

viii. I also made a mistake with not having an implies on the power topology quality attributes defined outside the generalized topology.
ix. I also made a mistake with not including quality attributes for the motorInlineDP connector.
x. **Observation:** I have made a decent number of mistakes because I was missing values for certain quality attributes. It would be good to have some sort of checker to warn users about unconstrained integers.
    1. **Mitigation:** Static analysis to produce warning about possibly missing quality attribute values.
    2. **Classification:** Requested Feature/Possible Improvement
xi. I also removed the summations (totalCost, totalMass, and totalWarrantyCost) because I was not getting any solutions. With those removed I was able to find about 15,000 solutions which is not equal to what I got from plain Clafer, thus requires more debugging.

71. At this point I still have not finished debugging the generation, however, Eldar made lots of fixes to the generators as well as some other bugs. Additionally, he implemented a new way of modeling the deployment. This came about after some discussion and issues that were going to arise with how we originally had deployment modeled and it also allows us to slice up the layers completely for debugging purposes. So at this point I have gotten the changes and I need to make updates to the model. Before getting to update the list of bugs, I wanted to tackle the new deployment concepts.
    a. Again, like in previous steps, I started with the two door window. I first removed all deployment that I had from the FAA first.
    b. Once the old deployment was removed from the FAA I had to think about how the Clafer was going to be generated for the two door case, since I still had errors, I couldn't view the plain Clafer. What I was trying to see is that if I could write an abstract deployment but be able to restrict the elements it applied to for each instantiated system such as the driver and passenger door. In plain Clafer, we are able to do this through references.
        i. **Observation**: I still see a pitfall here where we are still not restricting the instantiated deployment to apply the concrete systems of driver and passenger but rather to the abstract system. I think this will become very apparent when I try and generate Clafer.
            1. **Mitigation:** See 71.g
            2. **Classification:** See 71.g
    c. Moving forward, I first wrote the deployment expressions for the non-pinch detection FAA. This was quite simple, I just used the autocomplete menu to select a "Deploy" construct for each of my previous deployment and gave the same source and targets as I had earlier.
        i. **Bug**: "Deployment of" is missing as a valid option in the target of the "Deploy" construct.
    d. Once I had the base functions deployment modeled, the next thing I did was created a sub-deployment for the pinch detection functions. Here I chose PinchDetectionFA and

WinSysHA as my FAA and HA for the deployment. I then proceed to add the necessary deployment rues for the two functions in PinchDetectionFA.

i. Bug: You can't nest a "Deployment" under another "Deployment".
ii. Also I had to make the deployment optional as well as write an iff constraint similar to the one we had for the functional analysis architecture.
iii. Note: This will not be correct until we can nest the pinch detection deployment under the main deployment.

e. I still had to keep the constraints dictating the load power topology configuration.
f. Next I had to instantiate the deployment in the concrete systems. What I had previously where I gave the specific FAA and HA to the specific system deployment and inherited from the abstract one (the one we just created) was actually "okay".
g. **Observation**: This is still not going to work when we generate the plain Clafer. This is because we have no way of refining the references of FA and HA which are assigned by the generators to the specific values we want. Additionally, I found it a little unintuitive what the statement "Deployment DWinSysDpl of DWinSysFA to DWinSysHA of WinSysDpl" is actually doing. I think it would be better to explicitly redefine the references (even though plain Clafer does not support this yet for the chocosolver backend). I propose the following:
    i. We have a syntax "Deployment <some name> of <some optional super node>"
    ii. Then inside, using the intention menu we have two references, "functional analysis architecture" and "hardware architecture".
    iii. Next, when we do have reference refinement, the user could explicitly set the references in both places and maybe have some sort "refine" keyword that gets inserted when we refine the references in concrete deployments.
    iv. For now though, since we don't have it. We could relax the syntax and not require a user to set the references such that they only have to set them in there concrete deployments.
    v. **Mitigation:** Implement the above steps
    vi. **Classification:** Possible Pitfall/Bug
h. So at this point, deployment will still not work in the generated plain Clafer but we get no errors regarding deployment in MPS.

72. Since I have now updated the deployment, the next thing I needed to do was address the various errors that are in the model. It seemed like most of the errors were from the quality attributes. I got a lot of errors like "reference <some QA> (clafer) is out of search scope". Updating the references (using the autocomplete menu) fixed these errors.
    a. **Observation**: Not sure if this was a bug or not. Maybe a migration script can be used in the future?
        i. **Mitigation:** Migration script to aid with common errors that crop up due to reference model changes.
        ii. **Classification:** Feature Request/Possible Improvement
    b. **Observation**: As I was going through the quality attributes I realized one of my summations was incorrect, I was summing a mass and a cost.
        i. **Mitigation:** Static analysis to find mistakes in quality attribute equations or summations.

ii. **Classification:** Requested Feature/Possible Improvement

c. After updating these quality attribute references it removed all the errors in my model.

73. I then proceed to update the "DriverDoor" model based on the numerous changes I had already applied to the two door case.
    a. I first started with the FAA and removed the deployment and moved it to the deployment layer. I followed the same procedure as step 71.
        i. Note: This will not be correct until we can nest the pinch detection deployment under the main deployment.
    b. Next I moved to the hardware architecture.
        i. First I updated the references in the QA's to remove the errors.
        ii. Next I added the missing latency related quality attributes to the two modules and the hardware architecture.
        iii. Bug: The quality attribute table layout is super buggy.
    c. Lastly I updated the quality attributes for the system.
    d. At this point the "DriverDoor" model has been updated to be at the same stage as the two door model.
74. I removed the quality attributes pertaining to the timing chains for the two door model. I never had them in the driver door model.
75. At this point both power window models are at the same point and have the new deployment changes, while they be incorrect. The next thing I did was update the list of bugs that Eldar fixed after verifying that they were indeed fixed.
    a. Bug 58.f.iv was fixed so I was able to constrain the buses for the logical bus bridge in the two door model.
        i. **Bug**: Also with this bug fixed I was able write my transferTimePerSize equation. However, I get a warning: "Warning: argument of WHEN CONCRETE block is never concrete".
76. Now that I have my models up to date I want to generate Clafer but I know it will be incorrect due to the issues with deployment.
    a. After some discussion we felt that what was best was to not have a generic deployment. We can't have a generic deployment because of the issues that arise when you inherit a deployment and the need for reference refinement.
    b. I started with the DriverDoor model. I removed the old deployment and created a concrete deployment directly in DriverWinSys. Also I removed the unneeded imports from each of the modules in the model.
    c. Next I wanted to test the generators so I built the DriverDoor model and got the generated Clafer for the DriverWinSys.
        i. Bug: missing "deployedTo"
        ii. So there is still an issue with nesting the deployments inside of each other.
        iii. Bug: Always putting "this" before "fa" in the deployment causes problems when nesting under other clafers. We have to think of a smarter way to generate.
        iv. At this point I am able to compile but when I try and generate instances I am getting 0 so I need to debug. I am going to start by just looking at the functional analysis architecture with the feature model.

v. I can't just generate the layers individually right out of the gate. I have to make them now concrete instead of abstract.
vi. The functional analysis architecture and the feature model are good. Now I am going to try and bring in the hardware architecture. I would like to just bring in the device node classification but it is not split apart. To do this I am going to keep what I have and just paste in the hardware architecture.
vii. After uncommenting the commented parts for the reference model and making all the layers concrete I was able to generate 480 solutions. To ensure that this was correct, I compared did a similar thing for the plain Clafer model. This was tedious because I had all the quality attributes. The layers were pretty easy to comment out actually since I used a very modular style of writing my model.
viii. After comparing and getting way more instances for the plain Clafer approach I realized I forgot about of variability in my model. I then reviewed my model and found I was missing making all my hardware connectors optional and the BCM reference.
ix. I made these fixes and generated solutions and got closer within 1000. I figured again I was restricting variability in some way I didn't want to.
x. I had a hunch that it was the power topology causing the problems and it was… I stated previously it would have been nice to test it by itself to get my logic worked out. It turns out that…. that I made a mistake in my ClaferMPS implementation… I use DoorModule instead of DoorInline as a source in my "MotorIsDriver" configuration. I should have payed closer attention to my diagram layouts!
xi. I then looked closer at the diagrams and found another mistake in the DoorModuleIsDriver topology! After making these fixes, I regenerated the clafer and found the correct number of instances without considering deployment. When considering deployment I got an incorrect number.
xii. Due to the bug in the deployment still being present (70.h.ii.5) I was getting the wrong number of instances. After fixing this in manually in the reference model I was able to generate the correct number of instances when including all layers.
xiii. The last thing I needed to consider was making sure the quality attributes was correct. This I tested that I could get some instances based on queries that were relatively close but I did not expect to get the same as in plain Clafer due to how we model quality attributes at each layer in MPS and rounding issues that would arise.

d. At this point the DriverDoor model is **complete** and generates an equivelant Clafer to the plain Clafer approach and the reasoning can be done.
e. Now I wanted to move on to the TwoDoor Model and ensure that I could generate instances for it as well that were correct given possible issues with namespace. I first started by creating a specific deployment for both doors.
f. Before testing the generated Clafer I wanted to experiment with the idea of module testing. I felt that since we could generate modules independently we could test relatively small parts of the model (i.e., the power topology). To do this I created a new

module (in the two door model since I nicely split up the modules) called "Test_WinSysPT".

i. Observation: I would really like to just be able to instantiate a concrete power topology (outside of the system) for testing purposes but it is not possible at the moment.
ii. Thus I had to create a semi-complete stub system which instantiated the power topology (and the device node classification).
iii. I then wrote a set of constraints inside the content of my power topology. I mixed in plain Clafer in order to divide up the "test cases". The one test case I built was to test the logic that if I don't have any device power wires leaving the door inline then I should not provide device power to the inline.
iv. I then generated the clafer for the module Test_WinSysPT and made minor fixes (due to generator bugs) and was able to verify that I didn't get an unsat core thus satisfying the test case.

g. I then wanted to generate both the driver and passenger systems together. There is no good way of doing this so I just copied and pasted both modules into a single module with two systems name TwoDoorSystem. I then tried to build the model was able to.
h. I then got the generated Clafer and tried to compile it and generate instances.
    i. I first fixed the known bugs before compiling from when I tested the single door.
i. There was a big problem with namespaces. *Elaborate*.

77. Eldar then changed the DSL syntax to allow using expressions for connector expressions so that a user could specify the full path of the component they were looking for. This change allowed for me to fix issues with the bus bridge on the two door system where it was completely ambiguous which "localLowSpeedBus" I was talking about.
    a. I also made the paths explicit for the front passenger discrete connectors.
    b. I then generated the Clafer, fixed the known bugs but left any ambiguous naming alone. I got an ambiguous name so what I did was go back to MPS and update the ambiguous endpoints to use a full path.
    c. I removed WinSysHA because we have an issue where we can' generalize since the children are always concrete and you can't inherit from a concrete clafer.
78. As a result of this limitation and among other limitations we had another discussion and implemented the use of references in the topologies and deployment. However, the generated code is a hack currently because the choco backend does not support reference refinement.
    a. Using the new version of MPS I went back and modified the two door power window generalized architectures. For the generic communication topology, power topology, and hardware architecture I prefixed all device nodes with the "dn" reference.
    b. I then created two new generalized modules WinSysHA and WinSysDpl. I had these both before but due to the limitations I had to remove them. I added the HA back in and set the references to the generalized components. In the deployment I moved the common core components out and prefixed the FA and DN components with "fa." And "ha.dn." respectively.
    c. With this refactoring the MPS implementation is starting to look very much like our plain Clafer (in terms of structure). Also I believe most of the name spaces issues are resolved.
    d. The last thing I did was changed the BCM and EC references to use constraints to set their values since the generator was still not working correctly.

e. With this I then generated the Clafer for DriverWinSys and it should compile right away.
    i. I ran into a couple issues but they were related to pathing I used incorrectly in MPS.
    ii. Other than that the Clafer compiled correctly. I then went through and inspected it manually
    iii. I then generated the Clafer and got less instances than I was expecting.
    iv. …

79. I then applied the same refactoring to the single door model.
    a. Bug: There generation is not working correctly due to an ambiguous path with the reference “dn” for communication topology.
80. Eldar fixed the problem with references such that I can model the two door power window in the same manner.
    a. I applied the refactoring to both the single door and the two door model and tested a few cases to make sure I got the same instances for the two door model. For the driver I was able to just generate all instances and check that way.
        i. Observation: there is a corner case with the workaround with references where it might think there is an element accessible by the path but it is not because the reference is set as a set expression constraint.
        ii. The two door model takes way longer to find an instance than in the plain Clafer approach. We sort of expect this because we don’t guide what the limited scope for the allowed connectors are.
        iii. I found out that you don’t need the TwoDoorWinsys module since the FrontPassWinSys imports the DriverWinSys, thus it brings it in to the generated Clafer.
        iv. Bug: The logical door bus is missing the endpoint constraint.