The first thing you have to know is that the new FaMa format is easy to use, and easy to learn, please take a look into this tutorial to know how easy it is. The second thing you have to know is we mark 3 subsections into the format, %Relationships for the relations, %Attributes to specifies the attributes of the previously used features, and the %Constraint to specifies the cross-tree constraints and the complex constraints that we now support.

## Relationships

At first we have to put the line %Relationships to mark what we are going to write.

The way we creates the relations, is, origin : destination, At first line we put the root feature. If any destination is inside [] means it’s optional, or if you prefer you can put the cardinality inside [] before the name of the destination feature.

Examples:

Mandatoy relationship, between A and B , been A the root feature:

|  |
| --- |
| % Relationships  A : B ; |

|  |
| --- |
| % Relationships  A : [1,1] B; |

Optional feature between B and C, been A the root feature:

|  |
| --- |
| % Relationships  A : B;  B: [C]; |

|  |
| --- |
| % Relationships  A : B;  B: [0,1]C; |

Set relation over C and D,E:

|  |
| --- |
| C : [1,2] {D E} ; |

## Attributes

To write attributes we use the header %Attributes, for example, if we want to describe the attribute version of C.

|  |
| --- |
| % Attributes  C.version:[2,1,5],1,2; |

At first we specifies the Integer values inside [], and separated by commas, the second value is the default value, and the last one is the null value.

Also, you can specify an integer range (or a set of ranges):

|  |
| --- |
| % Attributes  C.version: Integer[2 to 5][10 to 15],1,2; |

## Constraints

We support traditional crosstree constraints, and more complex constraints as we show:

To write complex constrains they should be written in this way:

|  |
| --- |
| James.cost == UserManagement.cost + WSInterface.cost + GUI.cost + Core.cost + Modules.cost; |

Or even this way:

|  |
| --- |
| Wifi.speed > 5 AND OS.version > 10 IMPLIES Bluetooth; |

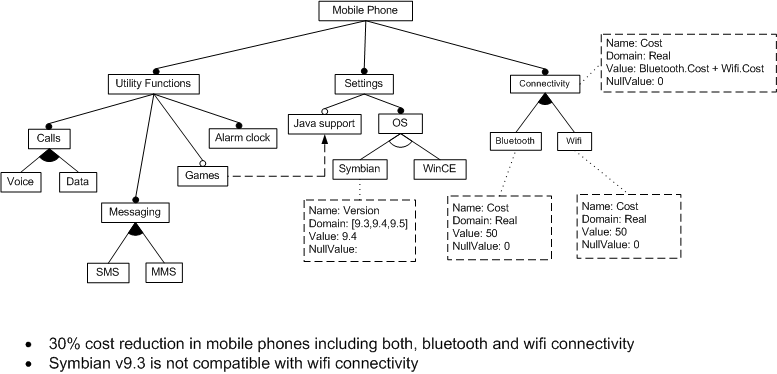
To write invariants:

|  |
| --- |
| Wifi{  speed > bandwich;  bandwith > 3 IMPLIES psicalLayer == 100;  } |

The tradicional relations should be written in this way:

|  |
| --- |
| CongressManagement REQUIRES Repository;  Repository EXCLUDES PDA; |

## Example:



|  |
| --- |
| **% Relationships**  MobilePhone : UtilityFunc Settings Connectivity;  UtilityFunc : Calls Messaging [Games] Alarm;  Calls : [1,2] {Voice Data};  Messaging : [1,2] {SMS MMS};  Settings : [JavaSupport] OS;  OS : [1,1] {Symbian WinCE};  Connectivity : [1,2] {Bluetooth Wifi};  **% Attributes**  Symbian.version: [0,3,4,5], 4, 0;  Bluetooth.cost: Integer[0 to 500], 50, 0;  Wifi.cost: Integer[0 to 50], 50, 0;  Connectivity.cost: [0,4],0 , 0;  **% Constraints and invariants**  Games IMPLIES JavaSupport;  Symbian.version==3 IMPLIES NOT wifi;  Bluetooth and Wifi IMPLIES Connectivity.cost= =(Bluetooth.cost +Wifi .cost);  Connectivity.cost == Bluetooth.cost + Wifi.cost; |