GÉANT - The GN3 Project
AutoBAHN
Stitching Prototype Development

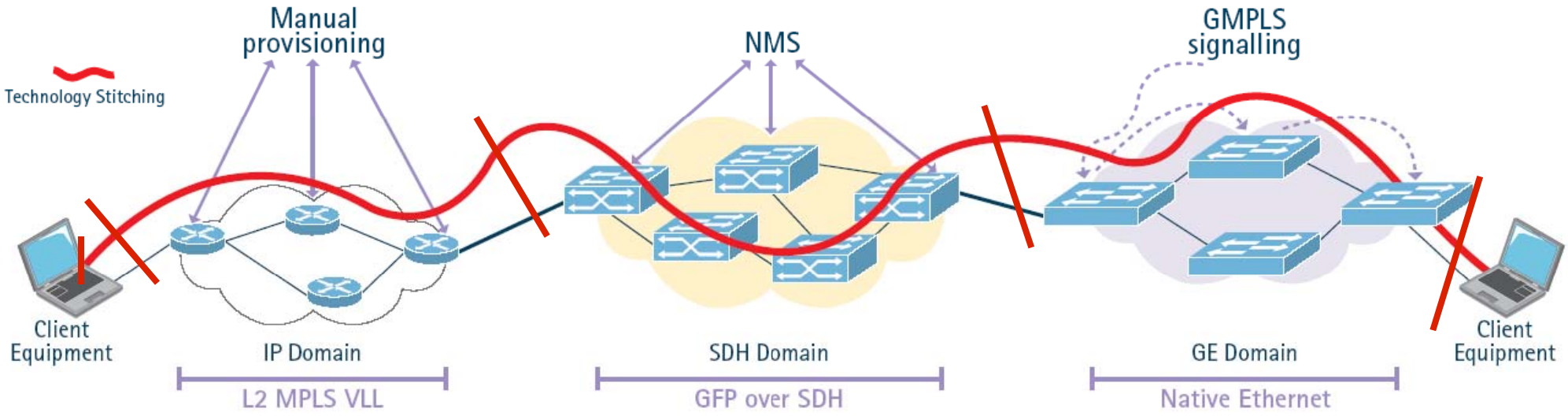
HEAnet Activity
Technology Stitching
Andrew Mackarel
Soren Krum

Suggested Agenda

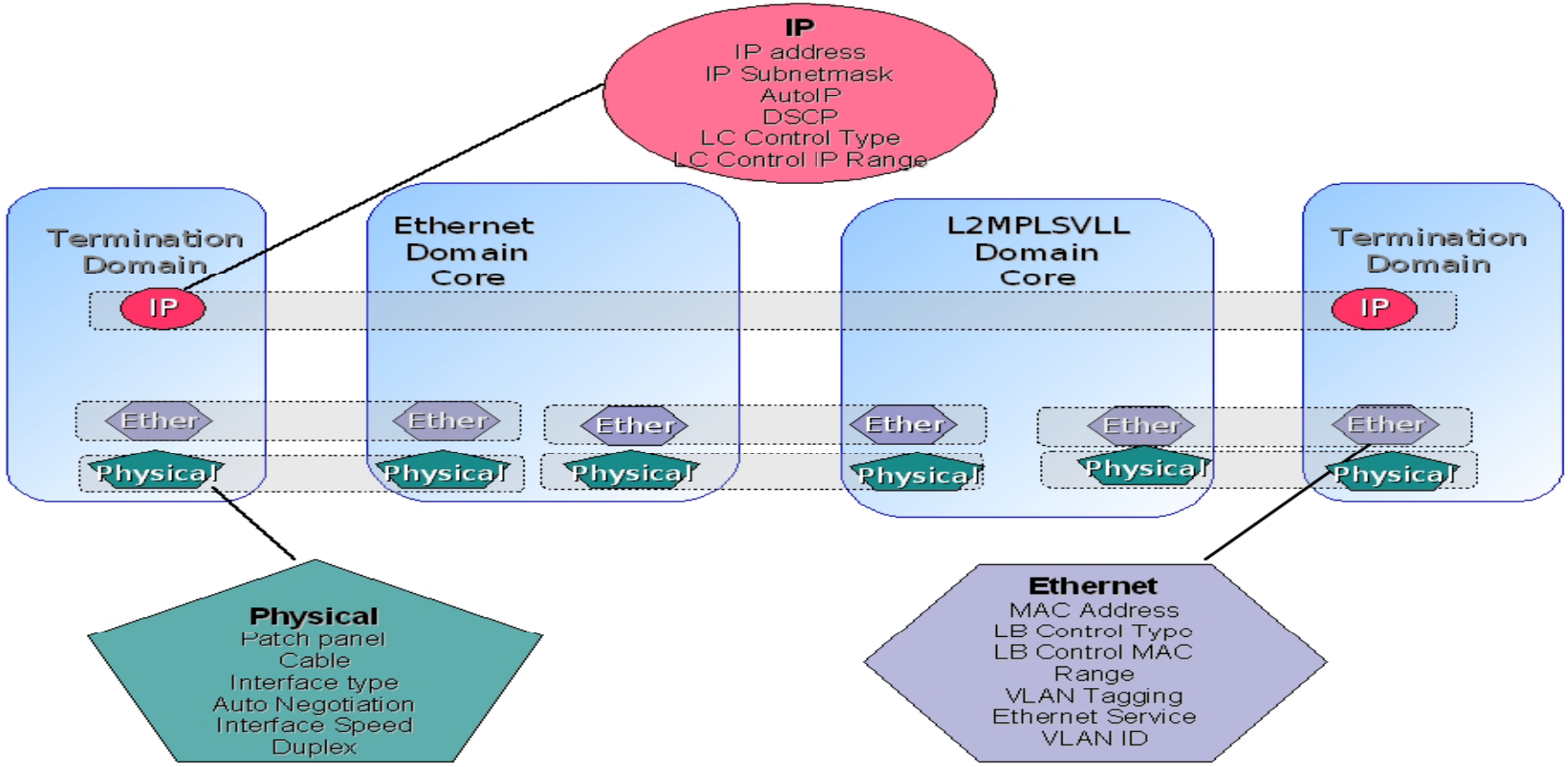
- Short review of stitching framework
- Discussion on architecture of stitching framework prototype implementation
  - What
  - Where
  - Which
  - How Many

Stitching within AutoBAHN

- Control and provisioning has to be distributed
- Business-layer related interactions include AA, policies, advance reservations etc. (All could be influenced by Stitching)



Stitching Technology



(Automating a process we all do manually everyday)

Stitching Framework

The stitching framework faces a problem:

It has to interact with

- A set of known parameters
- Although easily enhancable

Therefore:

- -> xml configuration (in spring)

stitching framework

Each domain has its own parameters, but how to interpret them?

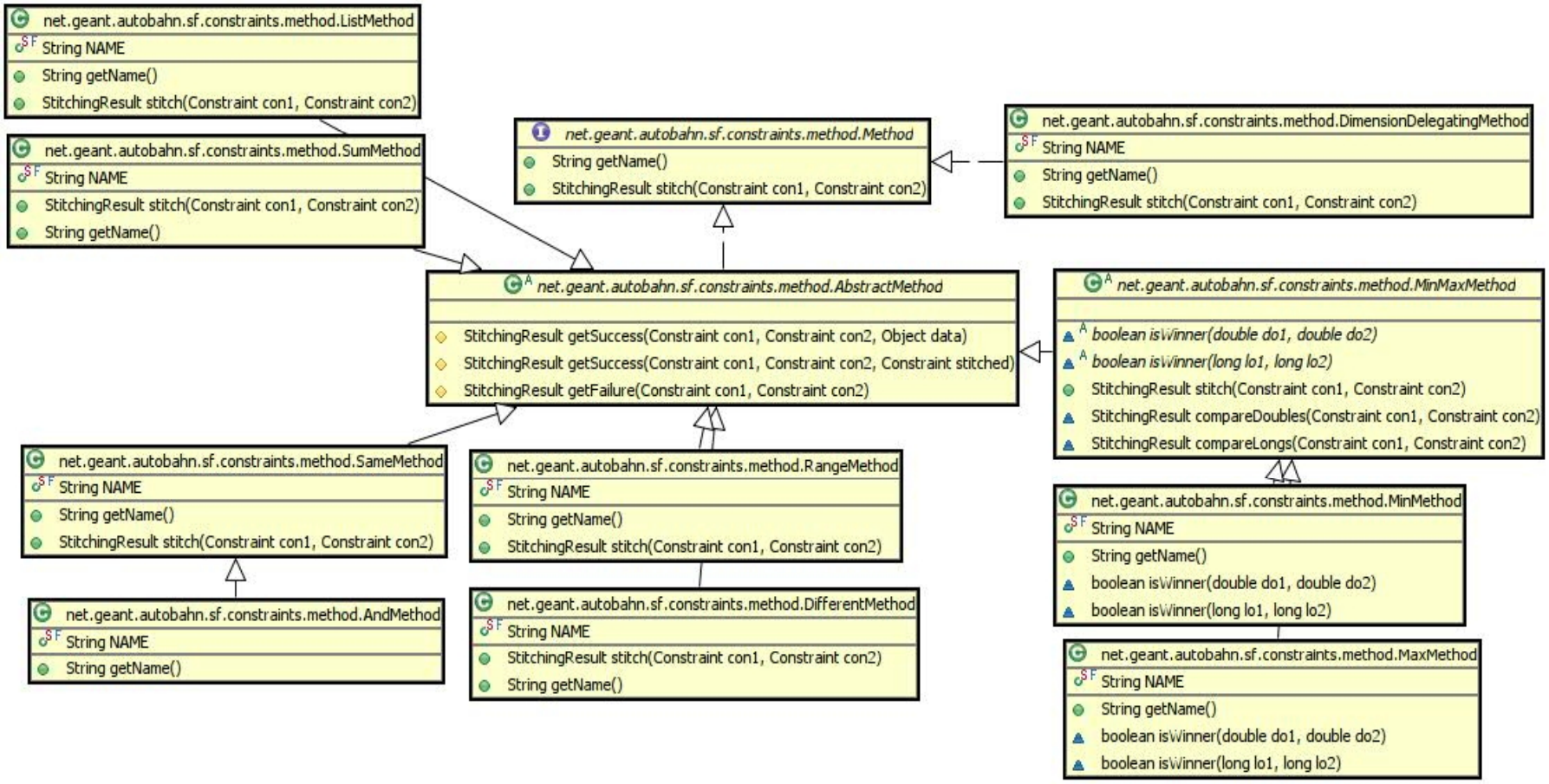
-> The Constraint class holds the attributes of a parameter

It holds information about

- What to do with a parameter
- Where that shall be done
- Which parameter
- How many of this parameter
- (A default value, if that is useful)

what?

- The Method knows what to do with a parameter

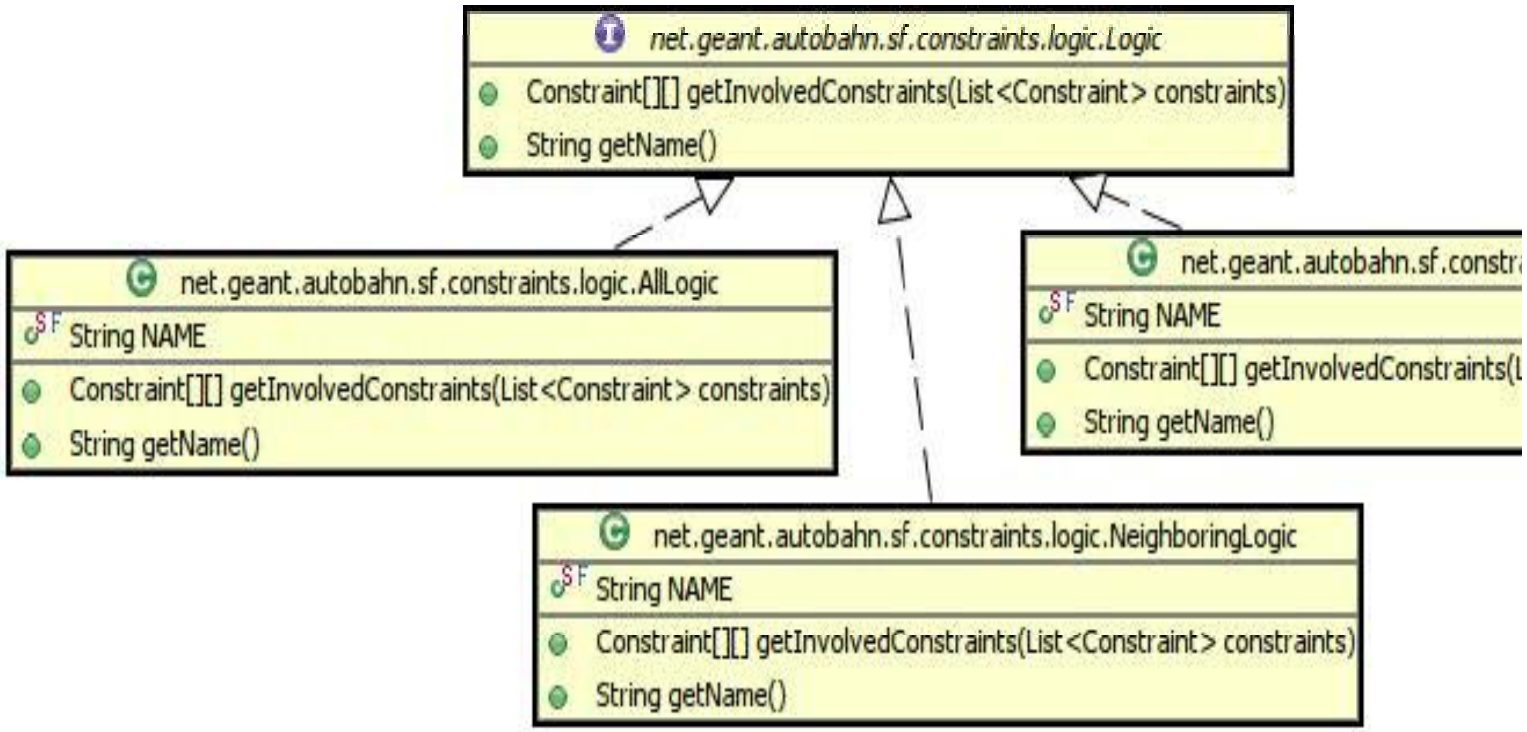


In xml...

- <bean id="andMethod" class="net.geant.autobahn.sf.constraints.method.AndMethod" />
- <bean id="dimensionDelegatingMethod" class="net.geant.autobahn.sf.constraints.method.DimensionDelegatingMethod" />
- <bean id="differentMethod" class="net.geant.autobahn.sf.constraints.method.DifferentMethod" />
- <bean id="listMethod" class="net.geant.autobahn.sf.constraints.method.ListMethod" />
- <bean id="maxMethod" class="net.geant.autobahn.sf.constraints.method.MaxMethod" />
- <bean id="minMethod" class="net.geant.autobahn.sf.constraints.method.MinMethod" />
- <bean id="rangeMethod" class="net.geant.autobahn.sf.constraints.method.RangeMethod" />
- <bean id="sameMethod" class="net.geant.autobahn.sf.constraints.method.SameMethod" />
- <bean id="sumMethod" class="net.geant.autobahn.sf.constraints.method.SumMethod" />

where?

- The logic knows where to do it

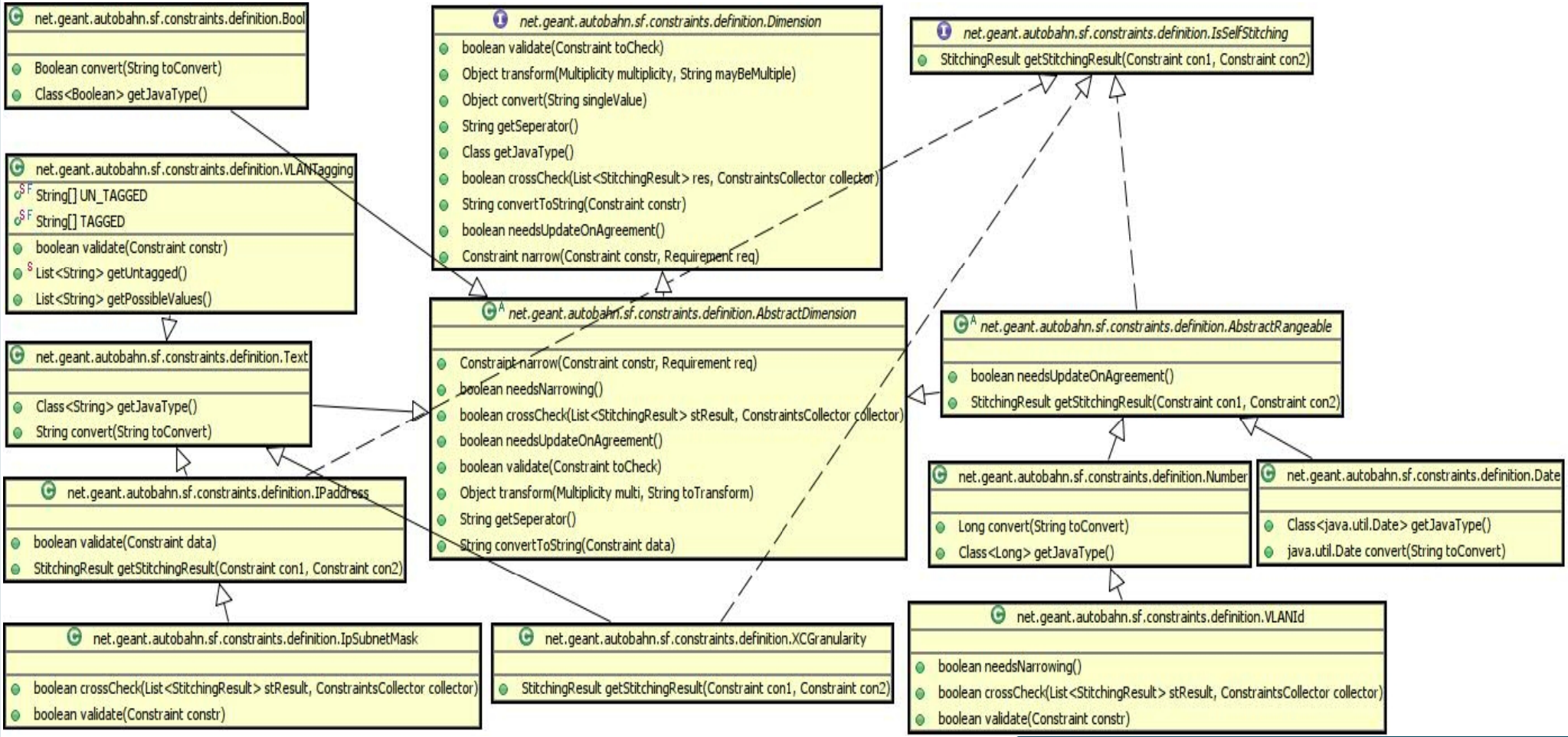


In xml...

```
"<bean id="allLogic" class="net.geant.autobahn.sf.constraints.logic.AllLogic" />
<bean id="pathLogic" class="net.geant.autobahn.sf.constraints.logic.PathLogic"
    />
<bean id="neighboringLogic"
    class="net.geant.autobahn.sf.constraints.logic.NeighboringLogic />
```

which?

- The dimension identifies the parameter



In xml...

```
<bean id="booleanDimension" class="net.geant.autobahn.sf.constraints.definition.Bool" />
<bean id="dateDimension" class="net.geant.autobahn.sf.constraints.definition.Date" />
<bean
    id="digitalNumberDimension"class="net.geant.autobahn.sf.constraints.definition.DigitalNumber"
    />
<bean id="ipAddressDimension" class="net.geant.autobahn.sf.constraints.definition.IPaddress" />
<bean id="ipSubnetMaskDimension"
    class="net.geant.autobahn.sf.constraints.definition.IpSubnetMask" />
<bean id="numberDimension" class="net.geant.autobahn.sf.constraints.definition.Number" />
<bean id="textDimension" class="net.geant.autobahn.sf.constraints.definition.Text" />
<bean id="vlanIdDimension" class="net.geant.autobahn.sf.constraints.definition.VLANId" />
<bean id="vlanTaggingDimension"
    class="net.geant.autobahn.sf.constraints.definition.VLANTagging" />
<bean id="xcGranularityDimension"
    class="net.geant.autobahn.sf.constraints.definition.XCGranularity" />
```

how many?

As that was seen as a closed case, there is one java class modelling this topic:

```
public enum Multiplicity {
    SINGLE ("single value"),
    LIST ("list of values"),
    RANGE("Range of values from the first to the second");
    private String name;
    private Multiplicity(final String name){ this.setName(name); }
    public String getName() {return name;}
    private void setName(final String name) { this.name = name;}
}
```

In xml...

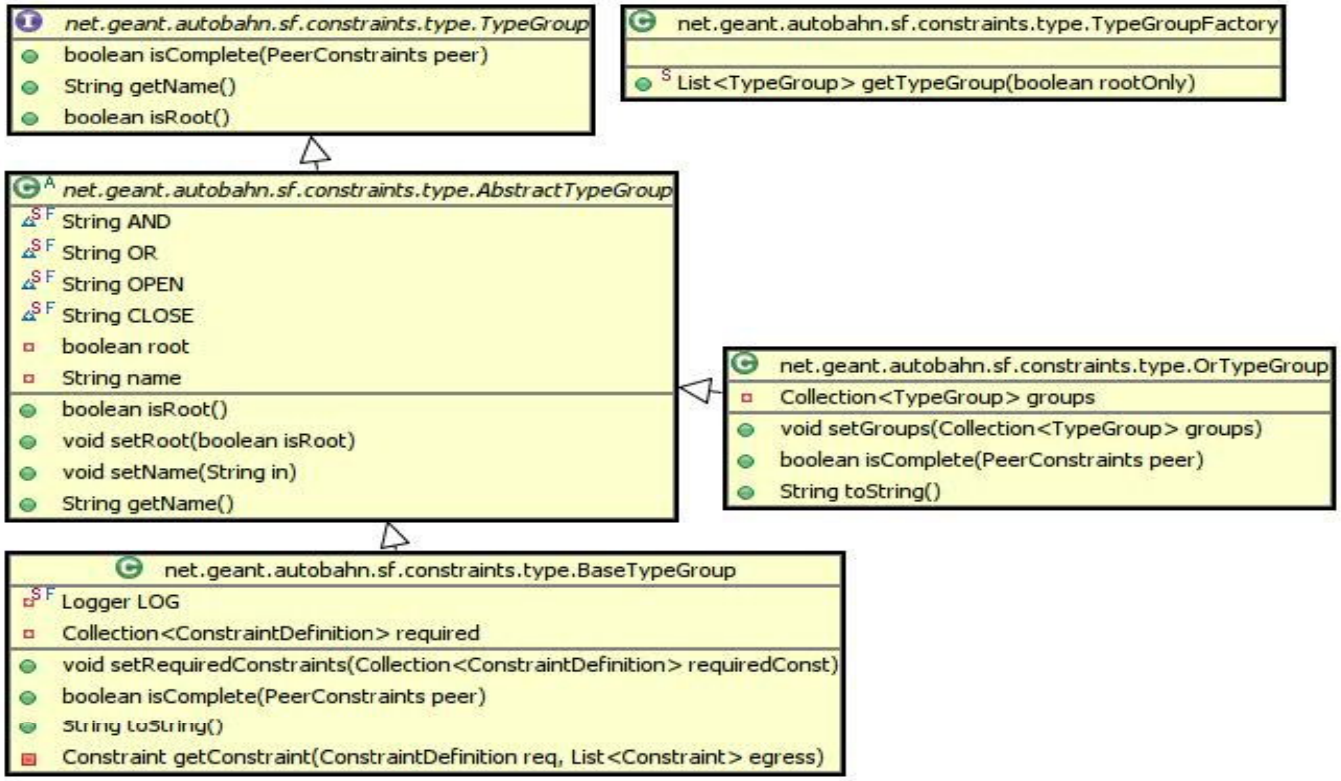
And here a constraint:

```
<bean id="IPSubnetMask"
    class="net.geant.autobahn.sf.constraints.ConstraintDefinition">
    <constructor-arg value="IPSubnetMask" />
    <constructor-arg ref="ipSubnetMaskDimension" />
    <constructor-arg ref="neighboringLogic" />
    <constructor-arg ref="sameMethod" />
    <constructor-arg value="SINGLE" />
    <property name="defaultValue" value="255.255.255.0" />
</bean>
```

A bit extra?

The Type...

... knows who should be present



how to add a new parameter

Imagine you figure out that cost is of interest on your path. What would you have to do?

- Add a cost constraint to your SF in constraintFactory.xml (and make it required...)
- Inform all InterdomainController to use the new Configuration
- Configure the InterdomainController, so they know how expensive their service is.

In xml ...

And here the constraint:

```
<bean id="cost" class="net.geant.autobahn.sf.constraints.ConstraintDefinition">
    <constructor-arg value="cost" /><!-- a name of your choice, but all IDC have to give
     that same name to their parameters -->
    <constructor-arg ref="digitalNumberDimension" /><!-- If you do not use digits, the
     numberDimension would be enough -->
    <constructor-arg ref="pathLogic" /><!-- ok, everyone wants its share -->
    <constructor-arg ref="sumMethod" /><!-- we want to add upp all values -->
    <constructor-arg value="SINGLE" /><!-- but they get it only one time -->
    <property name="defaultValue" value="20000" /><!-- we are not cheap by default ;-) -
    ->
</bean>
```

To be honest...

Ok, there is something missing in the last example:

-> The Currency!

But if you are in the happy Euro-zone(or any other place on this world using only one currency), it would work as easy!

Back up
Explanation Slides

Stitching data model

- All stitching parameter have the same attributes :
  - Name
  - Value
  - Intervention (tell who/what can change it)
    human, remote, auto, pass, none
  - Involved (which part of domain is involved)
    interface, whole domain
  - Dependency (dependency on other parameters)
  - Logic (the domains taking part in logic)
    peering, contiguous, pass, all
  - Method (how to compare values)
    same, different, min, max, function
- Attribute value can be a single value, a list, a range or a function
- Attribute value can be inherited (due to technology type of interface or core)

The stitching engine

- Parameters can be grouped for convenience
  e.g. because they belong to a layer. But layers are **not** essential.
- New parameters can be defined
  there are no boundaries to the number of parameters as long as the attributes are covered by the data model
- The stitching engine must be able to handle any number of parameters
- The stitching engine must be able to properly evaluate all attributes of the parameters in the path


GÉANT

In xml...

- And the Type:
- <bean id="cost_required" class="net.geant.autobahn.sf.constraints.type.BaseTypeGroup">
- <property name="root" value="true"/>
- <property name="name" value="cost_required"/>
- <property name="requiredConstraints">
- <set>
- <ref bean="cost"/>
- </set>
- </property>
- </bean>