**Topic**

**Topic**

**<u>Topic</u>**

- ☐ Code Reviewers
  - ☐ Testers
    - ☐ Karla
    - ☐ Kate
    - ☐ Chris
  - ☐ Commandos
    - ☐ Ilana Dashevky
    - ☐ Perry Rose
    - ☐ Jeff Stys
    - ☐ Ernie Morse
  - ☐ Developers
    - ☐ Andy
    - ☐ Tom
    - ☐ Pat - Moderator for Data Model, Reviewee for Schemas
    - ☐ Rob - Moderator for Schemas, Reviewee for Data Model

# APT Developer Primer

The following are some concepts and constructs used extensively in APT development that are not being reviewed, but should be understood in the context of the code review.

## Java Constructs

### Enums

Enums are a Java construct that allows explicitly named enumerated instances of a particular concept. They are often used in place of static int collections often seen in other programming languages. A Java Enum is a light-weight class, so it is fast to use, but also provides all the functionality of a class.

For more on Enums, there is a short description available here:

http://java.sun.com/j2se/1.5.0/docs/guide/language/enums.html

### Generics

Generics are a means of identifying the underlying classes contained by another class, and ensuring type safety at compile time. It is an alternative to explicit class casting done at run time, which is necessary to have access to class methods of the underlying object. In the case of Collections, this is a more object-oriented approach to providing the same functionality of a typed array.

With generics you *know* the class of the object you are working with at compile time. With casting, you *hope* you know the class of the object you are working with at run time.

For more on Generics, there is a short description available here:

http://java.sun.com/j2se/1.5.0/docs/guide/language/generics.html

## APT Team Libraries

### CoSI

The CoSI (Constraint Sequencing infrastructure) is a library of objects, and an infrastructure to support automatic rerunning of methods when dependent variables change. The basic design is that there are Properties, Constraints, and a Propagator.

The Propagator runs the Constraints. It has a queue of ones that are ready to run, and runs them sequentially. A Constraint is the equivalent of a method. It accesses some Properties, and sets others, all within the context of a single run() method. While the Constraint is running, all the Properties that it accesses track that this Constraint is dependent on that Property. When the Property changes, it tells the Propagator to add all dependent Constraints to its queue. This is a different way of handling property change events than the one normally supported by Java, but it allows a more declarative expression of the logic. This design was based on that used by Trans.

### Tina

Tina (Tina is no acronym) is a library of objects designed to support the proposal hierarchy. It is the basis for the HST and JWST document models. It also provides the GUI elements that are used for editing proposals.

The document model is divided into Elements and Fields. A TinaDocumentElement is a complex type that can be inserted in a document hierarchy in a parent child relationship with the document as a whole. TinaFields are more akin to attributes of a TinaDocumentElement. They represent a specific value and present a specific way of editing that one value. Tina provides all the infrastructure for managing the relationship between the Elements.

For the JWST project, we have begun to merge the Cosi Properties into the Tina Fields, allowing Tina Fields to be used to force Constraints to rerun.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns="http://www.stsci.edu/JWST/APT/Instrument/Nirspec"
    xmlns:jwst="http://www.stsci.edu/JWST/APT"
    targetNamespace="http://www.stsci.edu/JWST/APT/Instrument/Nirspec"
    elementFormDefault="qualified"
    version="4">

    <xs:import namespace="http://www.stsci.edu/JWST/APT" schemaLocation="../JwstDMSchema.xsd"/>

    <xs:complexType name="ReferenceStarsType">
        <xs:sequence>
            <xs:element name="ReferenceStar" type="ReferenceStarType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ReferenceStarType">
        <xs:sequence>
            <xs:element name="TargetID" type="xs:string" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>

    <xs:simpleType name="ReadoutPatternType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="NRS"/>
            <xs:enumeration value="NRSRAPID"/>
            <xs:enumeration value="NRSSLOW"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="SlitType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="S200A1"/>
            <xs:enumeration value="S200A2"/>
            <xs:enumeration value="S200B"/>
            <xs:enumeration value="S100A"/>
            <xs:enumeration value="S400A"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="SubarrayType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="DEFAULT"/>
            <xs:enumeration value="ALLSLITS"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="GratingType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="G140M"/>
            <xs:enumeration value="G235M"/>
            <xs:enumeration value="G395M"/>
            <xs:enumeration value="G140H"/>
            <xs:enumeration value="G235H"/>
            <xs:enumeration value="G395H"/>
            <xs:enumeration value="G140M"/>
            <xs:enumeration value="PRISM"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="FilterType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="F140X"/>
            <xs:enumeration value="F110W"/>
            <xs:enumeration value="F070LP"/>
            <xs:enumeration value="F100LP"/>
            <xs:enumeration value="F170LP"/>
            <xs:enumeration value="F290LP"/>
            <xs:enumeration value="F070LP"/>
            <xs:enumeration value="F100LP"/>
            <xs:enumeration value="F170LP"/>
            <xs:enumeration value="F290LP"/>
            <xs:enumeration value="CLEAR"/>
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns="http://www.stsci.edu/JWST/APT/Template/NirspecFixedSlitSpectroscopy"
    xmlns:nirspec="http://www.stsci.edu/JWST/APT/Instrument/Nirspec"
    targetNamespace="http://www.stsci.edu/JWST/APT/Template/NirspecFixedSlitSpectroscopy"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    version="1">

    <xs:import namespace="http://www.stsci.edu/JWST/APT/Instrument/Nirspec" schemaLocation="../../InstrumentSchemas/Nirspec.xsd"/>

    <xs:element name="NirspecFixedSlitSpectroscopy">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="AcqTargetID" type="xs:string" minOccurs="0"/>
                <xs:element name="AcqTargetFilter" type="nirspec:FilterType" minOccurs="0"/>
                <xs:element name="AcqReadoutPattern" type="nirspec:ReadoutPatternType" minOccurs="0"/>
                <xs:element name="MsaAcquisitionConfigFile" type="xs:string" minOccurs="0"/>
                <xs:element name="ReferenceStars" type="nirspec:ReferenceStarsType" minOccurs="0"/>
                <xs:element name="Slit" type="nirspec:SlitType" minOccurs="0"/>
                <xs:element name="Subarray" type="nirspec:SubarrayType" minOccurs="0"/>
                <xs:element name="Exposures" type="ExposuresType" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="ExposuresType">
        <xs:sequence>
            <xs:element name="Exposure" type="ExposureType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ExposureType">
        <xs:sequence>
            <xs:element name="Grating" type="nirspec:GratingType" minOccurs="0"/>
            <xs:element name="Filter" type="nirspec:FilterType" minOccurs="0"/>
            <xs:element name="ReadoutPattern" type="nirspec:ReadoutPatternType" minOccurs="0"/>
            <xs:element name="Groups" type="xs:int" minOccurs="0"/>
            <xs:element name="Integrations" type="xs:int" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>

</xs:schema>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns="http://www.stsci.edu/JWST/APT/Template/NirspecIFUSpectroscopy"
    xmlns:nirspec="http://www.stsci.edu/JWST/APT/Instrument/Nirspec"
    targetNamespace="http://www.stsci.edu/JWST/APT/Template/NirspecIFUSpectroscopy"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    version="1">

    <xs:import namespace="http://www.stsci.edu/JWST/APT/Instrument/Nirspec" schemaLocation="../../InstrumentSchemas/Nirspec.xsd"/>

    <xs:element name="NirspecIFUSpectroscopy">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="AcqTargetID" type="xs:string" minOccurs="0"/>
                <xs:element name="AcqTargetFilter" type="nirspec:FilterType" minOccurs="0"/>
                <xs:element name="AcqReadoutPattern" type="nirspec:ReadoutPatternType" minOccurs="0"/>
                <xs:element name="MsaAcquisitionConfigFile" type="xs:string" minOccurs="0"/>
                <xs:element name="ReferenceStars" type="nirspec:ReferenceStarsType" minOccurs="0"/>
                <xs:element name="Exposures" type="ExposuresType" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="ExposuresType">
        <xs:sequence>
            <xs:element name="Exposure" type="ExposureType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ExposureType">
        <xs:sequence>
            <xs:element name="Grating" type="nirspec:GratingType" minOccurs="0"/>
            <xs:element name="Filter" type="nirspec:FilterType" minOccurs="0"/>
            <xs:element name="ReadoutPattern" type="nirspec:ReadoutPatternType" minOccurs="0"/>
            <xs:element name="Groups" type="xs:int" minOccurs="0"/>
            <xs:element name="Integrations" type="xs:int" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>

</xs:schema>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns="http://www.stsci.edu/JWST/APT/Template/NirspecMOS"
    xmlns:nirspec="http://www.stsci.edu/JWST/APT/Instrument/Nirspec"
    targetNamespace="http://www.stsci.edu/JWST/APT/Template/NirspecMOS"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    version="1">

    <xs:import namespace="http://www.stsci.edu/JWST/APT/Instrument/Nirspec" schemaLocation="../../InstrumentSchemas/Nirspec.xsd"/>

    <xs:element name="NirspecMOS">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="IRImageAvailable" type="xs:boolean" minOccurs="0"/>
                <xs:element name="AcqTargetID" type="xs:string" minOccurs="0"/>
                <xs:element name="AcqTargetFilter" type="nirspec:FilterType" minOccurs="0"/>
                <xs:element name="AcqReadoutPattern" type="nirspec:ReadoutPatternType" minOccurs="0"/>
                <xs:element name="AcqMsaAcquisitionConfigFile" type="xs:string" minOccurs="0"/>
                <xs:element name="ReferenceStars" type="nirspec:ReferenceStarsType" minOccurs="0"/>

                <xs:element name="OptionalConfirmationImage" type="xs:boolean" minOccurs="0"/>
                <xs:element name="ConfirmationReadoutPattern" type="nirspec:ReadoutPatternType" minOccurs="0"/>
                <xs:element name="ConfirmationGroups" type="xs:int" minOccurs="0"/>

                <xs:element name="MsaAcquisitionConfigFile" type="xs:string" minOccurs="0"/>

                <xs:element name="Exposures" type="ExposuresType" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="ExposuresType">
        <xs:sequence>
            <xs:element name="Exposure" type="ExposureType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ExposureType">
        <xs:sequence>
            <xs:element name="Grating" type="nirspec:GratingType" minOccurs="0"/>
            <xs:element name="Filter" type="nirspec:FilterType" minOccurs="0"/>
            <xs:element name="ReadoutPattern" type="nirspec:ReadoutPatternType" minOccurs="0"/>
            <xs:element name="Groups" type="xs:int" minOccurs="0"/>
            <xs:element name="Integrations" type="xs:int" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>

</xs:schema>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns="http://www.stsci.edu/JWST/APT/Template/NirspecDark"
    xmlns:nirspec="http://www.stsci.edu/JWST/APT/Instrument/Nirspec"
    targetNamespace="http://www.stsci.edu/JWST/APT/Template/NirspecDark"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    version="4">

    <xs:import namespace="http://www.stsci.edu/JWST/APT/Instrument/Nirspec" schemaLocation="../../InstrumentSchemas/Nirspec.xsd"/>

    <xs:element name="NirspecDark">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Exposures" type="ExposuresType" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="ExposuresType">
        <xs:sequence>
            <xs:element name="Exposure" type="ExposureType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ExposureType">
        <xs:sequence>
            <xs:element name="Exposures" type="xs:int" minOccurs="0"/>
            <xs:element name="ReadoutPattern" type="nirspec:ReadoutPatternType" minOccurs="0"/>
            <xs:element name="Groups" type="xs:int" minOccurs="0"/>
            <xs:element name="Integrations" type="xs:int" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>

</xs:schema>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns="http://www.stsci.edu/JWST/APT/Template/NirspecFocus"
    xmlns:nirspec="http://www.stsci.edu/JWST/APT/Instrument/Nirspec"
    targetNamespace="http://www.stsci.edu/JWST/APT/Template/NirspecFocus"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    version="1">

    <xs:import namespace="http://www.stsci.edu/JWST/APT/Instrument/Nirspec" schemaLocation="../../InstrumentSchemas/Nirspec.xsd"/>

    <xs:element name="NirspecFocus">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="MsaAcquisitionConfigFile" type="xs:string" minOccurs="0"/>
                <xs:element name="TargetID" type="xs:string" minOccurs="0"/>
                <xs:element name="Filter" type="nirspec:FilterType" minOccurs="0"/>
                <xs:element name="ReadoutPattern" type="nirspec:ReadoutPatternType" minOccurs="0"/>
                <xs:element name="Groups" type="xs:int" minOccurs="0"/>
                <xs:element name="Integrations" type="xs:int" minOccurs="0"/>
                <xs:element name="RelativePositions" type="RelativePositions" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="RelativePositions">
        <xs:sequence>
            <xs:element name="RelativePosition" type="xs:int" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

</xs:schema>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns="http://www.stsci.edu/JWST/APT/Template/NirspecFocusReference"
    xmlns:nirspec="http://www.stsci.edu/JWST/APT/Instrument/Nirspec"
    targetNamespace="http://www.stsci.edu/JWST/APT/Template/NirspecFocusReference"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    version="1">

    <xs:import namespace="http://www.stsci.edu/JWST/APT/Instrument/Nirspec" schemaLocation="../../InstrumentSchemas/Nirspec.xsd"/>

    <xs:element name="NirspecFocusReference">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Direction" type="DirectionType" minOccurs="0"/>
                <xs:element name="Position" type="PositionType" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:simpleType name="DirectionType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="Forward"/>
            <xs:enumeration value="Reverse"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="PositionType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="MidStroke"/>
            <xs:enumeration value="Launch"/>
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns="http://www.stsci.edu/JWST/APT/Template/NirspecInternalLamp"
    xmlns:nirspec="http://www.stsci.edu/JWST/APT/Instrument/Nirspec"
    targetNamespace="http://www.stsci.edu/JWST/APT/Template/NirspecInternalLamp"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    version="1">

    <xs:import namespace="http://www.stsci.edu/JWST/APT/Instrument/Nirspec" schemaLocation="../../InstrumentSchemas/Nirspec.xsd"/>

    <xs:element name="NirspecInternalLamp">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="OperatingMode" type="OperatingMode" minOccurs="0"/>
                <xs:element name="MsaConfigFile" type="xs:string" minOccurs="0"/>
                <xs:element name="Exposures" type="xs:int" minOccurs="0"/>
                <xs:element name="ReadoutPattern" type="nirspec:ReadoutPatternType" minOccurs="0"/>
                <xs:element name="Groups" type="xs:int" minOccurs="0"/>
                <xs:element name="Integrations" type="xs:int" minOccurs="0"/>
                <xs:element name="Lamp" type="Lamp" minOccurs="0"/>
                <xs:element name="Grating" type="nirspec:GratingType" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:simpleType name="OperatingMode">
        <xs:restriction base="xs:string">
            <xs:enumeration value="MSASPEC"/>
            <xs:enumeration value="IFU"/>
            <xs:enumeration value="IMAGE"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="Lamp">
        <xs:restriction base="xs:string">
            <xs:enumeration value="FLAT1"/>
            <xs:enumeration value="FLAT2"/>
            <xs:enumeration value="FLAT3"/>
            <xs:enumeration value="FLAT4"/>
            <xs:enumeration value="FLAT5"/>
            <xs:enumeration value="LINE1"/>
            <xs:enumeration value="LINE2"/>
            <xs:enumeration value="LINE3"/>
            <xs:enumeration value="LINE4"/>
            <xs:enumeration value="REF"/>
            <xs:enumeration value="TEST"/>
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns="http://www.stsci.edu/JWST/APT/Template/NirspecMSAAnneal"
    xmlns:nirspec="http://www.stsci.edu/JWST/APT/Instrument/Nirspec"
    targetNamespace="http://www.stsci.edu/JWST/APT/Template/NirspecMSAAnneal"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    version="1">

    <xs:import namespace="http://www.stsci.edu/JWST/APT/Instrument/Nirspec" schemaLocation="../../InstrumentSchemas/Nirspec.xsd"/>

    <xs:element name="NirspecMSAAnneal">
        <xs:complexType>
            <xs:sequence>

            </xs:sequence>
        </xs:complexType>
    </xs:element>

</xs:schema>
```

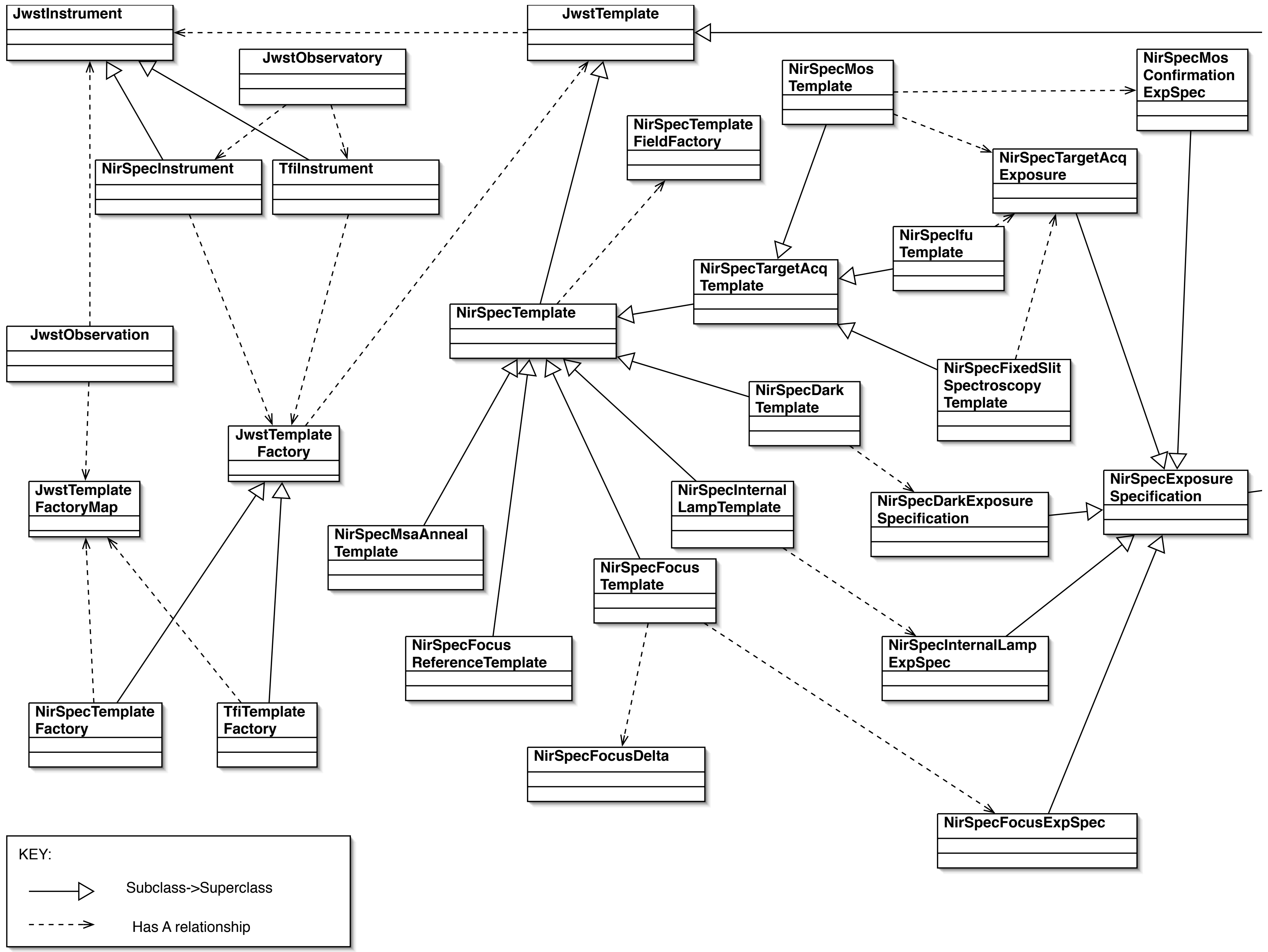
```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns="http://www.stsci.edu/JWST/APT/Instrument/Tfi"
    targetNamespace="http://www.stsci.edu/JWST/APT/Instrument/Tfi"
    elementFormDefault="qualified"
    version="1">

    <xs:simpleType name="SubarrayType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="FULL"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="ReadoutPatternType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="TFIG"/>
            <xs:enumeration value="TFIGRAPID"/>
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

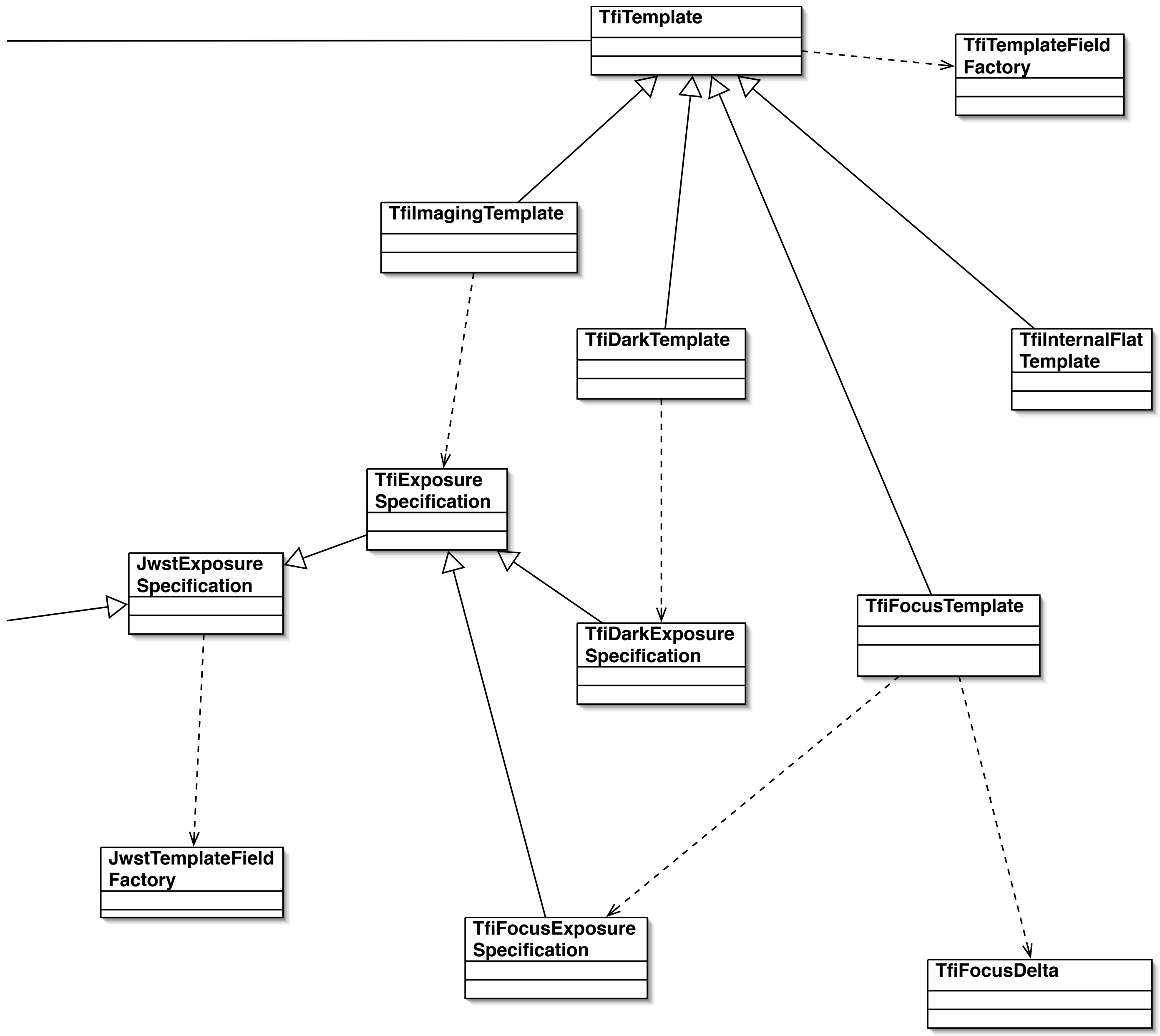
```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns="http://www.stsci.edu/JWST/APT/Template/TfiImaging"
    xmlns:jwst="http://www.stsci.edu/JWST/APT"
    xmlns:tfi="http://www.stsci.edu/JWST/APT/Instrument/Tfi"
    targetNamespace="http://www.stsci.edu/JWST/APT/Template/TfiImaging"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    version="1">

    <xs:import namespace="http://www.stsci.edu/JWST/APT" schemaLocation="../../JwstDMSchema.xsd"/>
    <xs:import namespace="http://www.stsci.edu/JWST/APT/Instrument/Tfi" schemaLocation="../../InstrumentSchemas/Tfi.xsd"/>

    <xs:element name="TfiImaging">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Subarray" type="tfi:SubarrayType" minOccurs="0"/>
                <xs:element name="ExposureList" type="ExposureListType" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="ExposureListType">
        <xs:sequence>
            <xs:element name="Exposure" type="ExposureType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ExposureType">
        <xs:sequence>
            <xs:element name="CentralWavelength" type="jwst:DoubleString" minOccurs="0"/>
            <xs:element name="MonochromaticImage" type="xs:boolean" minOccurs="0"/>
            <xs:element name="ReadoutPattern" type="tfi:ReadoutPatternType" minOccurs="0"/>
            <xs:element name="Groups" type="xs:int" minOccurs="0"/>
            <xs:element name="Integrations" type="xs:int" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>

</xs:schema>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns="http://www.stsci.edu/JWST/APT/Template/TfiDark"
    xmlns:tfi="http://www.stsci.edu/JWST/APT/Instrument/Tfi"
    targetNamespace="http://www.stsci.edu/JWST/APT/Template/TfiDark"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    version="1">

    <xs:import namespace="http://www.stsci.edu/JWST/APT/Instrument/Tfi" schemaLocation="../../InstrumentSchemas/Tfi.xsd"/>

    <xs:element name="TfiDark">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="ExposureList" type="ExposureListType" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="ExposureListType">
        <xs:sequence>
            <xs:element name="Exposure" type="ExposureType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ExposureType">
        <xs:sequence>
            <xs:element name="Exposures" type="xs:int" minOccurs="0"/>
            <xs:element name="ReadoutPattern" type="tfi:ReadoutPatternType" minOccurs="0"/>
            <xs:element name="Groups" type="xs:int" minOccurs="0"/>
            <xs:element name="Integrations" type="xs:int" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>

</xs:schema>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns="http://www.stsci.edu/JWST/APT/Template/TfiFocus"
    xmlns:jwst="http://www.stsci.edu/JWST/APT"
    xmlns:tfi="http://www.stsci.edu/JWST/APT/Instrument/Tfi"
    targetNamespace="http://www.stsci.edu/JWST/APT/Template/TfiFocus"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    version="1">

    <xs:import namespace="http://www.stsci.edu/JWST/APT" schemaLocation="../../JwstDMSchema.xsd"/>
    <xs:import namespace="http://www.stsci.edu/JWST/APT/Instrument/Tfi" schemaLocation="../../InstrumentSchemas/Tfi.xsd"/>

    <xs:element name="TfiFocus">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="TargetID" type="xs:string" minOccurs="0"/>
                <xs:element name="Pupil" type="PupilType" minOccurs="0"/>
                <xs:element name="WavelengthRange" type="WavelengthRangeType" minOccurs="0"/>
                <xs:element name="ReadoutPattern" type="tfi:ReadoutPatternType" minOccurs="0"/>
                <xs:element name="Groups" type="xs:int" minOccurs="0"/>
                <xs:element name="Integrations" type="xs:int" minOccurs="0"/>
                <xs:element name="RelativePositionList" type="RelativePositionListType" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:complexType name="RelativePositionListType">
        <xs:sequence>
            <xs:element name="RelativePosition" type="xs:double" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:simpleType name="PupilType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="CLEAR"/>
            <xs:enumeration value="MASK26"/>
            <xs:enumeration value="MASK31"/>
            <xs:enumeration value="MASK39"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="WavelengthRangeType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="SHORT"/>
            <xs:enumeration value="LONG"/>
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns="http://www.stsci.edu/JWST/APT/Template/TfiFlatSuite"
    targetNamespace="http://www.stsci.edu/JWST/APT/Template/TfiFlatSuite"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    version="1">

    <xs:element name="TfiFlatSuite">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="FlatSuite" type="FlatSuiteType" minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:simpleType name="FlatSuiteType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="PFLAT"/>
            <xs:enumeration value="LFLAT"/>
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```


JwstInstrument
JwstTemplate
JwstObservatory
NirSpecMos Template
NirSpecMos Confirmation ExpSpec
NirSpecTemplate FieldFactory
NirSpecInstrument
TfiInstrument
NirSpecTargetAcq Exposure
NirSpecIfu Template
NirSpecTargetAcq Template
NirSpecTemplate
JwstObservation
NirSpecFixedSlit Spectroscopy Template
NirSpecDark Template
JwstTemplate Factory
JwstTemplate FactoryMap
NirSpecInternal LampTemplate
NirSpecExposure Specification
NirSpecDarkExposure Specification
NirSpecMsaAnneal Template
NirSpecFocus Template
NirSpecFocus ReferenceTemplate
NirSpecInternalLamp ExpSpec
NirSpecTemplate Factory
TfiTemplate Factory
NirSpecFocusDelta
NirSpecFocusExpSpec
KEY:
Subclass->Superclass
Has A relationship


TfiTemplate
TfiTemplateField Factory
TfiImagingTemplate
TfiDarkTemplate
TfiInternalFlat Template
TfiExposure Specification
JwstExposure Specification
TfiDarkExposure Specification
TfiFocusTemplate
JwstTemplateField Factory
TfiFocusExposure Specification
TfiFocusDelta

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.instrument;

import java.util.Arrays;
import java.util.List;

/**
 * JwstObservatory - represents the actual telescope, contains the instruments and any
 * parameters we need to know about the Observatory itself.
 *
 * @author Rob Douglas
 */
public class JwstObservatory {
    //--------------------Statics--------------------
    private static final List<JwstInstrument> instruments = Arrays.asList(new JwstInstrument[]{
            MiriInstrument.getInstance(),
            NirCamInstrument.getInstance(),
            NirSpecInstrument.getInstance(),
            TfiInstrument.getInstance()
    });

    //--------------------Constructors--------------------
    private JwstObservatory (){}

    //--------------------Static Accessors--------------------
    public static List<JwstInstrument> getInstruments () {
        return instruments;
    }
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.instrument;

import java.util.List;

import edu.stsci.jwst.apt.model.template.JwstFilter;
/**
 * JwstInstrument - Defines what must be true about all instruments in JWST.
 *
 * @author Rob Douglas
 */
public abstract class JwstInstrument {

    private final String name;

    //--------------------Constructors--------------------

    public JwstInstrument(String iName) {
        name = iName;
        registerTemplates();
    }

    //--------------------Accessors--------------------
    public String getName() {
        return name;
    }

    public abstract List<? extends JwstFilter> getAcqFilters();

    //--------------------Methods--------------------

    public abstract double computeOverheadTime ();

    protected abstract void registerTemplates ();

    @Override
    public String toString() {
        return getName();
    }
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.instrument;

import java.util.Arrays;
import java.util.List;

import edu.stsci.jwst.apt.model.template.JwstFilter;
import edu.stsci.jwst.apt.model.template.JwstReadoutPattern;
import edu.stsci.jwst.apt.model.template.JwstTemplateFactoryMap;
import edu.stsci.jwst.apt.model.template.NirSpecTemplateFactory;

/**
 * NirSpecInstrument - represents the NIRSpec and all its associated data parameter values that
 * can be set for NIRSpec.
 *
 * @author Rob Douglas
 */
public class NirSpecInstrument extends JwstInstrument {
    //--------------------Statics--------------------
    private static final NirSpecInstrument NIRSPEC_INSTRUMENT = new NirSpecInstrument();

    public static final NirSpecFilter[] ACQ_FILTERS = new NirSpecFilter[] {
        NirSpecFilter.F140X,
        NirSpecFilter.F110W
    };

    //--------------------Constructors--------------------
    //Singleton
    private NirSpecInstrument (){
        super("NIRSPEC");
    }

    //--------------------Static Accessors--------------------
    public static NirSpecInstrument getInstance() {
        return NIRSPEC_INSTRUMENT;
    }

    //--------------------Accessors--------------------
    @Override
    public List<NirSpecFilter> getAcqFilters() {
        return Arrays.asList(ACQ_FILTERS);
    }

    //--------------------Methods--------------------
    @Override
    public double computeOverheadTime() {
        throw new UnsupportedOperationException();
    }

    @Override
    protected void registerTemplates() {
        for (NirSpecTemplateFactory iFactory : NirSpecTemplateFactory.values()) {
            JwstTemplateFactoryMap.registerFactoriesForInstrument(this, iFactory);
        }
    }

    //[start]--------------------Enums--------------------
    public enum NirSpecSlit {
        S200A1,
        S200A2,
        S200B,
        S100A,
        S400A
    }

    public enum NirSpecFilter implements JwstFilter {
        F140X,
        F110W,
        F070LP,
        F100LP,
        F170LP,
        F290LP,
        CLEAR
    }

    public enum NirSpecGrating {
        G140M,
        G235M,
        G395M,
        G140H,
        G235H,
        G395H,
        PRISM
    }

    public enum NirSpecGratingFilter {
        G140M_F070LP("G140M/F070LP", NirSpecGrating.G140M, NirSpecFilter.F070LP),
        G140M_F100LP("G140M/F100LP", NirSpecGrating.G140M, NirSpecFilter.F100LP),
        G235M_F170LP("G235M/F170LP", NirSpecGrating.G235M, NirSpecFilter.F170LP),
        G395M_F290LP("G395M/F290LP", NirSpecGrating.G395M, NirSpecFilter.F290LP),
        G140H_F070LP("G140H/F070LP", NirSpecGrating.G140H, NirSpecFilter.F070LP),
        G140H_F100LP("G140H/F100LP", NirSpecGrating.G140H, NirSpecFilter.F100LP),
        G235H_F170LP("G235H/F170LP", NirSpecGrating.G235H, NirSpecFilter.F170LP),
        G395H_F290LP("G395H/F290LP", NirSpecGrating.G395H, NirSpecFilter.F290LP),
        PRISM_CLEAR("PRISM/CLEAR", NirSpecGrating.PRISM, NirSpecFilter.CLEAR);

        private final String gratingFilterName;
        private final NirSpecGrating grating;
        private final NirSpecFilter filter;
        private NirSpecGratingFilter(String iName, NirSpecGrating iGrating, NirSpecFilter iFilter) {
            gratingFilterName = iName;
            grating = iGrating;
            filter = iFilter;
        }
        public String getGratingFilterName() {
            return gratingFilterName;
        }
        public NirSpecGrating getGrating() {
            return grating;
        }
        public NirSpecFilter getFilter() {
            return filter;
        }
        @Override
        public String toString () {
            return gratingFilterName;
        }
    }

    public enum NirSpecSubarray {
        DEFAULT,
        ALLSLITS
    }

    public enum NirSpecReadoutPattern implements JwstReadoutPattern {
        NRS(250),
        NRSRAPID(30),
        NRSSLOW(50);

        private final int maxGroups;
        private NirSpecReadoutPattern(int iMaxGroups) {
            maxGroups = iMaxGroups;
        }
        public int getMaxGroups () {
            return maxGroups;
        }
    }
    public enum NirSpecDirection {
        FORWARD,
        REVERSE
    }

    public enum NirSpecPosition {
        MID_STROKE,
        LAUNCH
    }

    public enum NirSpecOpMode {
        MSASPEC,
        IFU,
        IMAGE
    }

    public enum NirSpecLamp {
        FLAT1,
        FLAT2,
        FLAT3,
        FLAT4,
        FLAT5,
        LINE1,
        LINE2,
        LINE3,
        LINE4,
        REF,
        TEST
    }
    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.instrument;

import java.util.Arrays;
import java.util.List;

import edu.stsci.jwst.apt.model.template.JwstFilter;
import edu.stsci.jwst.apt.model.template.JwstReadoutPattern;
import edu.stsci.jwst.apt.model.template.JwstTemplateFactoryMap;
import edu.stsci.jwst.apt.model.template.TfiTemplateFactory;

/**
 * TfiInstrument - represents the TFI and all its associated data parameter values that
 * can be set for TFI.
 *
 * @author Rob Douglas
 */
public class TfiInstrument extends JwstInstrument {

    //[start]--------------------Statics--------------------
    private static final TfiInstrument TFI_INSTRUMENT = new TfiInstrument();

    public static final TfiFilter[] ACQ_FILTERS = new TfiFilter[] {
    };
    //[end]

    //[start]--------------------Constructors--------------------
    //Singleton
    private TfiInstrument(){
        super("TFI");
    }
    //[end]

    //[start]--------------------Static Accessor--------------------
    public static TfiInstrument getInstance(){
        return TFI_INSTRUMENT;
    }
    //[end]

    //[start]--------------------Accessors--------------------
    @Override
    public List<TfiFilter> getAcqFilters() {
        return Arrays.asList(ACQ_FILTERS);
    }
    //[end]

    //[start]--------------------Methods--------------------
    @Override
    public double computeOverheadTime() {
        throw new UnsupportedOperationException();
    }

    @Override
    protected void registerTemplates() {
        for (TfiTemplateFactory iFactory : TfiTemplateFactory.values()) {
            JwstTemplateFactoryMap.registerFactoriesForInstrument(this, iFactory);
        }
    }
    //[end]

    //[start]--------------------Enums--------------------
    public enum TfiFilter implements JwstFilter {
        F157M(1.500, 1.596),
        F176M(1.597,1.844),
        F200M(1.845,2.122),
        F226M(2.123,2.377),
        F245M(2.378,2.579),
        F348M(3.080,3.495),
        F403M(3.496,4.512),
        F481M(4.513,5.000);

        private final double minWave;
        private final double maxWave;
        private TfiFilter(double iMinWave, double iMaxWave) {
            minWave = iMinWave;
            maxWave = iMaxWave;
        }

        //[start]--------------------Accessors-------------------
        public double getMinWave() {
            return minWave;
        }
        public double getMaxWave() {
            return maxWave;
        }
        //[end]
    }

    public enum TfiSubarray {
        FULL
    }

    public enum TfiReadoutPattern implements JwstReadoutPattern {
        TFIG(250),
        TFIGRAPID(30);

        private final int maxGroups;
        private TfiReadoutPattern(int iMaxGroups) {
            maxGroups = iMaxGroups;
        }
        public int getMaxGroups () {
            return maxGroups;
        }
    }

    public enum TfiFlatSuite {
        PFLAT,
        LFLAT
    }

    public enum TfiPupil {
        CLEAR,
        MASK26,
        MASK31,
        MASK39
    }

    public enum TfiWavelengthRange {
        SHORT,
        LONG
    }
    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template;

import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Vector;

import edu.stsci.jwst.apt.model.instrument.JwstInstrument;

/**
 * JwstTemplateFactory - a registry of all the JWST Templates which will control which ones to
 * allow with each instrument.
 *
 * @author Rob Douglas
 */
public final class JwstTemplateFactoryMap {
    //[start]--------------------Statics--------------------
    private static final Map<JwstInstrument, List<JwstTemplateFactory>> FACTORY_MAP =
        new HashMap<JwstInstrument, List<JwstTemplateFactory>>();
    //[end]

    //[start]--------------------Constructors--------------------
    private JwstTemplateFactoryMap(){}
    //[end]

    //[start]--------------------Methods--------------------
    /*
     * This method requires the instrument to be passed in, even though the Factory has a
     * getInstrument method.  The reason for this is a bootstrapping issue.  Since this is
     * called while an instrument is being constructed, the TemplateFactories won't yet have
     * a valid reference to their instruments.  The only known reference that can be relied
     * on is to use this within the Instrument that is being initialized.
     */
    public static void registerFactoriesForInstrument (JwstInstrument iInstrument, JwstTemplateFactory iFactory) {
        List<JwstTemplateFactory> lInstrumentTemplates = FACTORY_MAP.get(iInstrument);
        if (lInstrumentTemplates==null) {
            lInstrumentTemplates = new Vector<JwstTemplateFactory>();
            FACTORY_MAP.put(iInstrument, lInstrumentTemplates);
        }
        lInstrumentTemplates.add(iFactory);
    }

    public static List<JwstTemplateFactory> getFactoriesForInstrument(JwstInstrument iInstrument) {
        return FACTORY_MAP.get(iInstrument);
    }
    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template;

import edu.stsci.jwst.apt.model.instrument.JwstInstrument;
import edu.stsci.tina.model.TinaFactory;

/**
 * JwstTemplateType - an interface to collect all the templates for JWST.
 *
 * @author Rob Douglas
 *
 */
public interface JwstTemplateFactory extends TinaFactory<JwstTemplate<? extends JwstInstrument>> {
    //[start]--------------------Accessors--------------------
    public String getTemplateName();
    public JwstInstrument getInstrument();
    //[end]

    //[start]--------------------Methods--------------------
    public JwstTemplate<? extends JwstInstrument> makeInstance();
    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template;

import edu.stsci.jwst.apt.model.instrument.JwstInstrument;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument;
import edu.stsci.jwst.apt.model.template.nirspec.*;

/**
 * NirSpecTemplateType - The templates for the NIRSPec instrument
 *
 * @author Rob Douglas
 */
public enum NirSpecTemplateFactory implements JwstTemplateFactory {
    NIRSPEC_FIXED_SLIT_SPEC("NIRSpec Fixed Slit Spectroscopy") {
        @Override
        public NirSpecFixedSlitTemplate makeInstance() {
            return new NirSpecFixedSlitTemplate();
        }
    },
    NIRSPEC_IFU_SPEC("NIRSpec IFU Spectroscopy") {
        @Override
        public NirSpecIfuTemplate makeInstance() {
            return new NirSpecIfuTemplate();
        }
    },
    NIRSPEC_MULTIOBJECT_SPEC("NIRSpec MultiObject Spectroscopy") {
        @Override
        public NirSpecMosTemplate makeInstance() {
            return new NirSpecMosTemplate();
        }
    },
    NIRSPEC_DARK("NIRSpec Dark"){
        @Override
        public NirSpecDarkTemplate makeInstance() {
            return new NirSpecDarkTemplate();
        }
    },
    NIRSPEC_FOCUS("NIRSpec Focus"){
        @Override
        public NirSpecFocusTemplate makeInstance() {
            return new NirSpecFocusTemplate();
        }
    },
    NIRSPEC_FOCUS_REF("NIRSpec Focus Reference"){
        @Override
        public NirSpecFocusReferenceTemplate makeInstance() {
            return new NirSpecFocusReferenceTemplate();
        }
    },
    NIRSPEC_INTERNAL_LAMP("NIRSpec Internal Lamp"){
        @Override
        public NirSpecInternalLampTemplate makeInstance() {
            return new NirSpecInternalLampTemplate();
        }
    },
    NIRSPEC_MSA_ANNEAL("NIRSpec MSA Anneal"){
        @Override
        public NirSpecMsaAnnealTemplate makeInstance() {
            return new NirSpecMsaAnnealTemplate();
        }
    };

    //[start]--------------------Fields--------------------
    private final String templateName;
    private final JwstInstrument instrument = NirSpecInstrument.getInstance();
    //[end]
    //[start]--------------------Constructors--------------------
    NirSpecTemplateFactory(String iTemplateName){
        this.templateName = iTemplateName;
    }
    //[end]

    //[start]--------------------Accessors--------------------
    public String getTemplateName() {return templateName;}

    public JwstInstrument getInstrument() {return instrument;}
    //[end]

    //[start]--------------------Methods--------------------
    public abstract JwstTemplate<? extends JwstInstrument> makeInstance();

    @Override
    public String toString() {
        return getTemplateName();
    }
    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template;

import edu.stsci.jwst.apt.model.instrument.JwstInstrument;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument;
import edu.stsci.jwst.apt.model.template.tfi.TfiDarkTemplate;
import edu.stsci.jwst.apt.model.template.tfi.TfiFocusTemplate;
import edu.stsci.jwst.apt.model.template.tfi.TfiImagingTemplate;
import edu.stsci.jwst.apt.model.template.tfi.TfiInternalFlatTemplate;

/**
 * TfiTemplateType - The templates for the Tfi instrument
 *
 * @author Rob Douglas
 */
public enum TfiTemplateFactory implements JwstTemplateFactory {
    TFI_IMAGING("TFI Imaging") {
        @Override
        public TfiImagingTemplate makeInstance() {
            return new TfiImagingTemplate();
        }
    },
    TFI_DARK("TFI Dark"){
        @Override
        public TfiDarkTemplate makeInstance() {
            return new TfiDarkTemplate();
        }
    },
    TFI_FOCUS("TFI Focus"){
        @Override
        public TfiFocusTemplate makeInstance() {
            return new TfiFocusTemplate();
        }
    },
    TFI_INTERNAL_FLAT("TFI Internal Flat"){
        @Override
        public TfiInternalFlatTemplate makeInstance() {
            return new TfiInternalFlatTemplate();
        }
    };

    //[start]--------------------Fields--------------------
    private final String templateName;
    private final JwstInstrument instrument = TfiInstrument.getInstance();
    //[end]

    //[start]--------------------Constructors--------------------
    TfiTemplateFactory(String iTemplateName){
        this.templateName = iTemplateName;
    }
    //[end]

    //[start]--------------------Accessors--------------------
    public String getTemplateName() {return templateName;}

    public JwstInstrument getInstrument() {return instrument;}
    //[end]

    //[start]--------------------Methods--------------------
    public abstract JwstTemplate<? extends JwstInstrument> makeInstance();

    @Override
    public String toString() {
        return getTemplateName();
    }
    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template;

import edu.stsci.tina.model.TinaDocumentElement;
import edu.stsci.tina.model.fields.CosiConstrainedDouble;
import edu.stsci.tina.model.fields.CosiConstrainedInt;

/**
 * @author Rob Douglas
 *
 */
public abstract class JwstTemplateFieldFactory {
    //[start]--------------------Statics----------------------
    public static final String READOUT_PATTERN = "Readout Pattern";
    public static final String REQUESTED_EXPOSURE_TIME = "Requested ExpTime";
    public static final String NUMBER_OF_EXPS = "No. of Exposures";
    public static final String NUMBER_OF_GROUPS = "No. of Groups";
    public static final String NUMBER_OF_INTS = "No. of Integrations";
    public static final String ACTUAL_EXP_TIME = "Actual Exp Time";

    //[end]
    //[start]--------------------Fields-----------------------


    //[end]
    //[start]--------------------Constructors-----------------


    //[end]
    //[start]--------------------Accessors--------------------


    //[end]
    //[start]--------------------Methods----------------------
    public static CosiConstrainedDouble makeRequestedExpTimeField (TinaDocumentElement iContainer) {
        return new CosiConstrainedDouble(iContainer, REQUESTED_EXPOSURE_TIME, true, 0.0, null);
    }

    public static CosiConstrainedInt makeNumberOfExpsField (TinaDocumentElement iContainer) {
        CosiConstrainedInt lNumExps = new CosiConstrainedInt(iContainer, NUMBER_OF_EXPS, true, 1, null);
        lNumExps.set(1);
        return lNumExps;
    }

    public static CosiConstrainedInt makeNumberOfGroupsField (TinaDocumentElement iContainer) {
        CosiConstrainedInt lNumGroups = new CosiConstrainedInt(iContainer, NUMBER_OF_GROUPS, true, 1, null);
        lNumGroups.set(1);
        return lNumGroups;
    }

    public static CosiConstrainedInt makeNumberOfIntsField (TinaDocumentElement iContainer) {
        CosiConstrainedInt lNumInts = new CosiConstrainedInt(iContainer, NUMBER_OF_INTS, true, 1, null);
        lNumInts.set(1);
        return lNumInts;
    }

    public static CosiConstrainedDouble makeActualExpTimeField (TinaDocumentElement iContainer) {
        CosiConstrainedDouble lField =
            new CosiConstrainedDouble(iContainer, ACTUAL_EXP_TIME, true, 0.0, null);
        lField.setEditable(false);
        return lField;
    }
    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template;

/**
 * @author Rob Douglas
 *
 */
public interface JwstFilter {

}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template;

/**
 * @author Rob Douglas
 *
 */
public interface JwstReadoutPattern {

}
```

```
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template;

/**
 * @author Rob Douglas
 *
 */
import org.jdom.Element;

import edu.stsci.CoSI.CosiConstraint;
import edu.stsci.CoSI.CosiDerivedProperty;
import edu.stsci.jwst.apt.view.JwstExposureSpecificationFormBuilder;
import edu.stsci.tina.form.FormFactory;
import edu.stsci.tina.model.AbstractTinaDocumentElement;
import edu.stsci.tina.model.TinaField;
import edu.stsci.tina.model.fields.CosiConstrainedDouble;
import edu.stsci.tina.model.fields.CosiConstrainedInt;

public abstract class JwstExposureSpecification extends AbstractTinaDocumentElement {

    static {
        FormFactory.registerFormBuilder(JwstExposureSpecification.class, new JwstExposureSpecificationFormBuilder());
    }

    //[start]--------------------Statics----------------------
    //[end]
    //[start]--------------------Fields-----------------------
    //TODO ROB These should not be public, but should be shared with the packages below template, but we
    //need superpackages for that, so for now, they have to be public
    public final CosiConstrainedDouble requestedExpTime = JwstTemplateFieldFactory.makeRequestedExpTimeField(this);
    public final CosiConstrainedInt numberOfGroupsField = JwstTemplateFieldFactory.makeNumberOfGroupsField(this);;
    public final CosiConstrainedInt numberOfIntegrationsField= JwstTemplateFieldFactory.makeNumberOfIntsField(this);
    public final CosiConstrainedDouble actualExpTimeField = JwstTemplateFieldFactory.makeActualExpTimeField(this);

    //TODO ROB Remove these generic shadow properties when CosiDerivedTinaField comes on-line
    protected CosiDerivedProperty<Integer> numberOfGroups;
    protected CosiDerivedProperty<Integer> numberOfIntegrations;
    protected CosiDerivedProperty<Double> actualExpTime;
    //[end]
    //[start]--------------------Constructors-----------------
    public JwstExposureSpecification() {
        super();
    }
    //[end]
    //[start]--------------------Accessors--------------------
    public abstract JwstTemplate getTemplate();

    public Integer getNumberOfGroups() {
        return numberOfGroupsField.get();
    }

    public Integer getNumberOfIntegrations() {
        return numberOfIntegrationsField.get();
    }

    public abstract JwstReadoutPattern getReadoutPattern();

    public Double getActualExpTime() {
        return actualExpTime.get();
    }

    public void setNumberOfGroups(Integer iNumGroups) {
        numberOfGroupsField.set(iNumGroups);
    }

    public void setNumberOfGroups(String iNumGroups) {
        numberOfGroupsField.setValueFromString(iNumGroups);
    }

    public void setNumberOfIntegrations(Integer iIntegrations) {
        numberOfIntegrationsField.set(iIntegrations);
    }

    public void setNumberOfIntegrations(String iIntegrations) {
        numberOfIntegrationsField.setValueFromString(iIntegrations);
    }

    @Override
    public String getTypeName() {
        return "JWST Exposure";
    }

    public Element getDomElement() {
        throw new UnsupportedOperationException();
    }
    //[end]
    //[start]--------------------Methods----------------------
    @Override
    public String toString() {
        String lStringValue = getClass()+": ";
        for (TinaField lField : getProperties()) {
            lStringValue+=lField.getName()+"="+lField.getValue();
        }

        return lStringValue;
    }

    protected abstract void configureEditableFields();

    //[end]
    //[start]--------------------Constraints------------------
    @CosiConstraint
    private void cosiSetActualExpTime() {
        actualExpTimeField.set(actualExpTime.get());
    }
    //[end]
    //[start]--------------------Inner Classes----------------
    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template;

import org.jdom.Element;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.instrument.JwstInstrument;
import edu.stsci.jwst.apt.view.DefaultJwstFormBuilder;
import edu.stsci.tina.form.FormFactory;
import edu.stsci.tina.model.AbstractTinaDocumentElement;

/**
 * An abstract class defining what a Template is in JWST.  It needs to have an
 * associated instrument, and define its own properties.
 *
 * @author Rob Douglas
 */
public abstract class JwstTemplate<T extends JwstInstrument> extends AbstractTinaDocumentElement {

    static {
        FormFactory.registerFormBuilder(JwstTemplate.class, new DefaultJwstFormBuilder());
    }

    //--------------------Fields--------------------
    protected final T instrument;
    protected final boolean internal;

    //--------------------Constructors--------------------
    public JwstTemplate(T iInstrument) {
        this(iInstrument, false);
    }

    public JwstTemplate(T iInstrument, boolean iInternal) {
        instrument = iInstrument;
        internal = iInternal;
        Cosi.completeInitialization(this, JwstTemplate.class);
    }

    //--------------------Accessors--------------------

    /*
     * Returns the Instrument used by this Template
     */
    public T getInstrument() {
        return instrument;
    }

    public boolean isInternal() {
        return internal;
    }

    @Override
    public String getTypeName() {
        return "JWST Template";
    }

    public Element getDomElement() {
        throw new UnsupportedOperationException("DOM is not supported for JWST.");
    }

    public static boolean isOpgsTestMode() {
        return "true".equalsIgnoreCase(System.getProperty("opgs-test"));
    }

    //--------------------Methods--------------------
    /*
     * This method creates a new visit when one is needed.  It is used
     * by ensureVisits.
     */
//    public JwstVisit createVisit (TinaField[] iProperties) {
//        //Need to decipher iProperties to get the parameters that should be set on the visit
//        //For now, a blank visit is ok.
//        return new JwstVisit();
//    }

    /*
     * The method keeps track of visits and creates new ones when needed that
     * fit the required parameters.  This keeps visits from being created and
     * destroyed when different ones are needed due to visit expansion.
     */
//    public void ensureVisits (JwstObservation iObs, TinaField[] iProperties) {
//        JwstVisit lVisit = iObs.ensureVisit(this, iProperties);
//        iObs.add(lVisit, true);
//        lVisit.setEditable(false);
//    }

}
```

```
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import java.util.ArrayList;
import java.util.List;

import edu.stsci.jwst.apt.model.JwstReferenceStar;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.*;
import edu.stsci.jwst.apt.model.template.JwstTemplate;
import edu.stsci.jwst.apt.model.template.JwstTemplateFieldFactory;
import edu.stsci.tina.model.*;
import edu.stsci.tina.model.fields.*;

/**
 * NirSpecTemplateFieldFactory - Provides a standard way to access the properties of a NIRSpec
 * template that are shared among them.  Its only purpose is to return Fields that can be
 * used within the Templates.
 *
 * @author Rob Douglas
 */
public final class NirSpecTemplateFieldFactory extends JwstTemplateFieldFactory {
    //--------------------Statics--------------------
    public static final String IR_IMAGE_AVAIL = "IR Image Available?";
    public static final String CONFIRM_IMAGE = "Obtain Confirmation Image?";
    public static final String SLIT = "Slit";
    public static final String SUBARRAY = "Subarray";
    public static final String GRATING_FILTER = "Grating/Filter";
    public static final String EXPOSURES = "Exposures";
    public static final String FILTER = "Filter";
    public static final String OPMODE = "Operating Mode";
    public static final String LAMP = "Lamp";
    public static final String GRATING = "Grating";
    public static final String DIRECTION = "Direction";
    public static final String POSITION = "Position";
    public static final String MSA_FILE = "MSA File";
    public static final String REFERENCE_STARS = "Reference Stars";
    public static final String DELTA_ARRAY = "Relative Positions";
    public static final String DELTA_NUMBER = "Number";
    public static final int DELTA_NUM_MIN = 1;
    public static final int DELTA_NUM_MAX = 20;
    public static final String DELTA = "Relative Position";
    public static final int DELTA_MIN = -18400;
    public static final int DELTA_MAX = 18400;

    public static final List<TableColumn<? super NirSpecExposureSpecification, ?>> NIRSPEC_EXP_COLUMNS = new ArrayList<
TableColumn<? super NirSpecExposureSpecification, ?>>();
    static {
        NIRSPEC_EXP_COLUMNS.add(new NirSpecGratingFilterColumn());
        NIRSPEC_EXP_COLUMNS.add(new NirSpecReadOutPatternColumn());
        NIRSPEC_EXP_COLUMNS.add(new NirSpecNumOfGroupsColumn());
        NIRSPEC_EXP_COLUMNS.add(new NirSpecNumOfIntsColumn());
        NIRSPEC_EXP_COLUMNS.add(new NirSpecActualExpTimeColumn());
    }

    public static final RowFactory<NirSpecExposureSpecification> NIRSPEC_EXP_FACTORY = new RowFactory<
NirSpecExposureSpecification>(){
        public NirSpecExposureSpecification makeInstance(TinaColumnMultiField parent) {
            return new NirSpecExposureSpecification((NirSpecTemplate)parent.getContainer());
        }
    };

    public static final List<TableColumn<? super NirSpecDarkExposureSpecification, ?>> NIRSPEC_DARK_COLUMNS = new ArrayList<
TableColumn<? super NirSpecDarkExposureSpecification, ?>>();
    static {
        NIRSPEC_DARK_COLUMNS.add(new NirSpecDarkNumberOfExpColumn());
        NIRSPEC_DARK_COLUMNS.add(new NirSpecReadOutPatternColumn());
        NIRSPEC_DARK_COLUMNS.add(new NirSpecNumOfGroupsColumn());
        NIRSPEC_DARK_COLUMNS.add(new NirSpecNumOfIntsColumn());
        NIRSPEC_DARK_COLUMNS.add(new NirSpecActualExpTimeColumn());
    }

    public static final RowFactory<NirSpecDarkExposureSpecification> NIRSPEC_DARK_FACTORY = new RowFactory<
NirSpecDarkExposureSpecification>(){
        public NirSpecDarkExposureSpecification makeInstance(TinaColumnMultiField parent) {
            return new NirSpecDarkExposureSpecification((NirSpecDarkTemplate)parent.getContainer());
        }
    };

    public static final List<TableColumn<? super NirSpecFocusDelta, ?>> NIRSPEC_FOCUS_DELTA_COLUMNS = new ArrayList<
TableColumn<? super NirSpecFocusDelta, ?>>();
    static {
        NIRSPEC_FOCUS_DELTA_COLUMNS.add(new NirSpecFocusDeltaNumberColumn());
        NIRSPEC_FOCUS_DELTA_COLUMNS.add(new NirSpecFocusDeltaColumn());
    }

    public static final RowFactory<NirSpecFocusDelta> NIRSPEC_FOCUS_DELTA_FACTORY = new RowFactory<NirSpecFocusDelta>(){
        public NirSpecFocusDelta makeInstance(TinaColumnMultiField parent) {
            return new NirSpecFocusDelta(parent.getRowCount()+1);
        }
    };

    //--------------------Constructors--------------------
    //Singleton - this is a static only class
    private NirSpecTemplateFieldFactory() {}

    //--------------------Methods--------------------
    public static CosiConstrainedSelection<NirSpecSlit> makeSlitField (JwstTemplate<NirSpecInstrument> iIU) {
        return new CosiConstrainedSelectionBuilder<NirSpecSlit>(iIU, SLIT, true).setLegalValues(NirSpecSlit.values()).build();
    }

    public static CosiConstrainedSelection<NirSpecSubarray> makeSubarrayField (JwstTemplate<NirSpecInstrument> iIU) {
        return new CosiConstrainedSelectionBuilder<NirSpecSubarray>(iIU, SUBARRAY, true).setLegalValues(NirSpecSubarray.values
()).build();
    }

    public static CosiConstrainedSelection<NirSpecReadoutPattern> makeReadoutPatternField (TinaDocumentElement iContainer) {
        return makeReadoutPatternField(iContainer, READOUT_PATTERN);
    }

    public static CosiConstrainedSelection<NirSpecReadoutPattern> makeReadoutPatternField (TinaDocumentElement iContainer,
String iName) {
        return new CosiConstrainedSelectionBuilder<NirSpecReadoutPattern>(iContainer, iName, true).setLegalValues
(NirSpecReadoutPattern.values()).build();
    }

    public static CosiBooleanField makeIRImageAvailField(NirSpecMosTemplate iContainer) {
        CosiBooleanField lIRAvailField = new CosiBooleanField(iContainer, IR_IMAGE_AVAIL, true);
        lIRAvailField.setTrueWord("Yes");
        lIRAvailField.setFalseWord("No");
        return lIRAvailField;
    }

    public static CosiBooleanField makeConfirmImageField(NirSpecMosTemplate iContainer) {
        CosiBooleanField lConfirm = new CosiBooleanField(iContainer, CONFIRM_IMAGE, true);
        lConfirm.setTrueWord("Yes");
        lConfirm.setFalseWord("No");
        lConfirm.set(false);
        return lConfirm;
    }

    public static TinaCosiStringField makeMsaFileField(TinaDocumentElement iContainer) {
        return makeMsaFileField(iContainer, MSA_FILE);
    }

    public static TinaCosiStringField makeMsaFileField(TinaDocumentElement iContainer, String iName) {
        return new TinaCosiStringField(iContainer, iName, true);
    }

    public static CosiConstrainedMultiSelection<JwstReferenceStar> makeReferenceStarField (NirSpecTemplate iIU) {
        return new CosiConstrainedMultiSelection<JwstReferenceStar>(iIU, REFERENCE_STARS, true, 8, 20,
                new JwstReferenceStar.ReferenceStarUIDGenerator(), new JwstReferenceStar.ReferenceStarComparator());
    }

    public static CosiConstrainedSelection<NirSpecGratingFilter> makeGratingFilterField (TinaDocumentElement iIU) {
        return new CosiConstrainedSelectionBuilder<NirSpecGratingFilter>(iIU, GRATING_FILTER, true).setLegalValues
(NirSpecGratingFilter.values()).build();
    }

    public static CosiConstrainedSelection<NirSpecOpMode> makeOpmodeField (NirSpecInternalLampTemplate iIU) {
        return new CosiConstrainedSelectionBuilder<NirSpecOpMode>(iIU, OPMODE, true).setLegalValues(NirSpecOpMode.values()).
build();
    }

    public static CosiConstrainedSelection<NirSpecLamp> makeLampField (NirSpecInternalLampTemplate iIU) {
        return new CosiConstrainedSelectionBuilder<NirSpecLamp>(iIU, LAMP, true).setLegalValues(NirSpecLamp.values()).build();
    }

    public static CosiConstrainedSelection<NirSpecGrating> makeGratingField (NirSpecInternalLampTemplate iIU) {
        return new CosiConstrainedSelectionBuilder<NirSpecGrating>(iIU, GRATING, true).setLegalValues(NirSpecGrating.values())
.build();
    }

    public static CosiConstrainedSelection<NirSpecFilter> makeFilterField (TinaDocumentElement iContainer) {
        return makeFilterField(iContainer, FILTER);
    }

    public static CosiConstrainedSelection<NirSpecFilter> makeFilterField (TinaDocumentElement iContainer, String iName) {
        return new CosiConstrainedSelectionBuilder<NirSpecFilter>(iContainer, iName, true).setLegalValues(NirSpecFilter.values
()).build();
    }


    public static CosiConstrainedSelection<NirSpecDirection> makeDirectionField (NirSpecFocusReferenceTemplate iIU) {
        return new CosiConstrainedSelectionBuilder<NirSpecDirection>(iIU, DIRECTION, true).setLegalValues(NirSpecDirection.
values()).build();
    }

    public static CosiConstrainedSelection<NirSpecPosition> makePositionField (NirSpecFocusReferenceTemplate iIU) {
        return new CosiConstrainedSelectionBuilder<NirSpecPosition>(iIU, POSITION, true).setLegalValues(NirSpecPosition.values
()).build();
    }

    public static TinaColumnMultiField<NirSpecExposureSpecification> makeNirSpecExposureTableField (NirSpecTemplate iIU) {
        return new TinaColumnMultiField<NirSpecExposureSpecification>(iIU, EXPOSURES, null, NIRSPEC_EXP_COLUMNS,
NIRSPEC_EXP_FACTORY);
    }

    public static TinaColumnMultiField<NirSpecDarkExposureSpecification> makeNirSpecDarkExposureTableField
(NirSpecDarkTemplate iIU) {
        return new TinaColumnMultiField<NirSpecDarkExposureSpecification>(iIU, EXPOSURES, null, NIRSPEC_DARK_COLUMNS,
NIRSPEC_DARK_FACTORY);
    }

    public static TinaColumnMultiField<NirSpecFocusDelta> makeNirSpecFocusDeltaTableField (NirSpecFocusTemplate iIU) {
        return new TinaColumnMultiField<NirSpecFocusDelta>(iIU, DELTA_ARRAY, null, NIRSPEC_FOCUS_DELTA_COLUMNS,
NIRSPEC_FOCUS_DELTA_FACTORY);
    }

    //[start]--------------------Inner Classes--------------------

    //[start] NirSpec Exposure Column Definitions (Template Independent)
    private static class NirSpecGratingFilterColumn extends CosiConstrainedSelectionColumn<NirSpecExposureSpecification>{
        public NirSpecGratingFilterColumn() {
            super(NirSpecTemplateFieldFactory.GRATING_FILTER);
        }
        public CosiConstrainedSelection get(NirSpecExposureSpecification getFrom) {
            return getFrom.gratingFilter;
        }
    }

    private static class NirSpecReqExpTimeColumn extends CosiConstrainedDoubleColumn<NirSpecExposureSpecification>{
        public NirSpecReqExpTimeColumn() {
            super(NirSpecTemplateFieldFactory.REQUESTED_EXPOSURE_TIME);
        }
        public CosiConstrainedDouble get(NirSpecExposureSpecification getFrom) {
            return getFrom.requestedExpTime;
        }
    }

    private static class NirSpecReadOutPatternColumn extends CosiConstrainedSelectionColumn<NirSpecExposureSpecification>{
        public NirSpecReadOutPatternColumn() {
            super(NirSpecTemplateFieldFactory.READOUT_PATTERN);
        }
        public CosiConstrainedSelection get(NirSpecExposureSpecification getFrom) {
            return getFrom.readoutPatternField;
        }
    }

    private static class NirSpecNumOfGroupsColumn extends CosiConstrainedIntColumn<NirSpecExposureSpecification>{
        public NirSpecNumOfGroupsColumn() {
            super(NirSpecTemplateFieldFactory.NUMBER_OF_GROUPS);
        }
        public CosiConstrainedInt get(NirSpecExposureSpecification getFrom) {
            return getFrom.numberOfGroupsField;
        }
    }

    private static class NirSpecNumOfIntsColumn extends CosiConstrainedIntColumn<NirSpecExposureSpecification>{
        public NirSpecNumOfIntsColumn() {
            super(NirSpecTemplateFieldFactory.NUMBER_OF_INTS);
        }
        public CosiConstrainedInt get(NirSpecExposureSpecification getFrom) {
            return getFrom.numberOfIntegrationsField;
        }
    }

    private static class NirSpecActualExpTimeColumn extends CosiConstrainedDoubleColumn<NirSpecExposureSpecification>{
        public NirSpecActualExpTimeColumn() {
            super(NirSpecTemplateFieldFactory.ACTUAL_EXP_TIME);
        }
        public CosiConstrainedDouble get(NirSpecExposureSpecification getFrom) {
            return getFrom.actualExpTimeField;
        }
    }

    //[end] NirSpec Imaging Column Definitions

    //[start] NirSpec Dark Column Definitions
    private static class NirSpecDarkNumberOfExpColumn extends CosiConstrainedIntColumn<NirSpecDarkExposureSpecification>{
        public NirSpecDarkNumberOfExpColumn() {
            super(NirSpecTemplateFieldFactory.NUMBER_OF_EXPS);
        }
        public CosiConstrainedInt get(NirSpecDarkExposureSpecification getFrom) {
            return getFrom.numExps;
        }
    }
    //[end] NirSpec Dark Column Definitions

    //[start] NirSpec Focus Delta Column Definitions
    private static class NirSpecFocusDeltaNumberColumn extends CosiConstrainedIntColumn<NirSpecFocusDelta>{
        public NirSpecFocusDeltaNumberColumn() {
            super(NirSpecTemplateFieldFactory.DELTA_NUMBER);
        }
        public CosiConstrainedInt get(NirSpecFocusDelta getFrom) {
            return getFrom.number;
        }
    }

    private static class NirSpecFocusDeltaColumn extends CosiConstrainedIntColumn<NirSpecFocusDelta>{
        public NirSpecFocusDeltaColumn() {
            super(NirSpecTemplateFieldFactory.DELTA);
        }
        public CosiConstrainedInt get(NirSpecFocusDelta getFrom) {
            return getFrom.delta;
        }
    }
    //[end] NirSpec Focus Delta Column Definitions

    //[end]
}
```

```
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecSubarray;
import edu.stsci.jwst.apt.model.template.JwstTemplate;

/**
 * @author rob
 *
 */
public abstract class NirSpecTemplate extends JwstTemplate<NirSpecInstrument> {
    //[start]--------------------Statics----------------------


    //[end]
    //[start]--------------------Fields-----------------------


    //[end]
    //[start]--------------------Constructors-----------------
    public NirSpecTemplate() {
        this(false);
    }

    public NirSpecTemplate(boolean iInternal) {
        super(NirSpecInstrument.getInstance(), iInternal);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    public abstract NirSpecSubarray getSubarray();

    //[end]
    //[start]--------------------Methods----------------------


    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;


import java.util.Set;

import edu.stsci.CoSI.Cosi;
import edu.stsci.CoSI.CosiConstraint;
import edu.stsci.jwst.apt.model.JwstProposalSpecification;
import edu.stsci.jwst.apt.model.JwstReferenceStar;
import edu.stsci.jwst.apt.model.JwstTargets;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecFilter;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecReadoutPattern;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecSubarray;
import edu.stsci.jwst.apt.view.template.nirspec.NirSpecTargetAcqTemplateFormBuilder;
import edu.stsci.tina.form.FormFactory;
import edu.stsci.tina.model.TinaField;
import edu.stsci.tina.model.fields.CosiConstrainedMultiSelection;
import edu.stsci.tina.model.fields.TinaCosiStringField;
import edu.stsci.tina.model.fields.TinaCosiField.BrokenLinkException;

/**
 * NirSpecTargetAcqTemplate - This class encapsulates the properties shared by templates that
 * need to do a special target acquisition.
 *
 * @author Rob Douglas
 */
public abstract class NirSpecTargetAcqTemplate extends NirSpecTemplate {
    //[start]--------------------Statics--------------------
    public static final String ACQ_MSA = "Acq MSA File";
    static {
        FormFactory.registerFormBuilder(NirSpecTargetAcqTemplate.class, new NirSpecTargetAcqTemplateFormBuilder());
    }
    //[end]

    //[start]--------------------Fields--------------------
    protected final TinaCosiStringField acqMsaFile = NirSpecTemplateFieldFactory.makeMsaFileField(this, ACQ_MSA);
    protected final NirSpecTargetAcqExposure acqExposure = new NirSpecTargetAcqExposure(this);
    protected final CosiConstrainedMultiSelection<JwstReferenceStar> acqRefStars = NirSpecTemplateFieldFactory.
makeReferenceStarField(this);
    {
        setProperties(new TinaField[] {acqExposure.acqFilter, acqExposure.readoutPatternField,
                acqExposure.numberOfGroupsField, acqExposure.numberOfIntegrationsField,
                acqExposure.actualExpTimeField, acqMsaFile,
                acqRefStars});
    }
    //[end]

    //[start]--------------------Constructors--------------------
    public NirSpecTargetAcqTemplate() {
        super();
        add(acqExposure, false);
        Cosi.completeInitialization(this, NirSpecTargetAcqTemplate.class);
    }
    //[end]

    //[start]--------------------Accessors--------------------
    public NirSpecTargetAcqExposure getAcqExposure() {
        return acqExposure;
    }

    public NirSpecFilter getAcqFilter() {
        return acqExposure.acqFilter.get();
    }

    public void setAcqFilter (NirSpecFilter iFilter) {
        acqExposure.acqFilter.set(iFilter);
    }

    public NirSpecReadoutPattern getAcqPattern() {
        return acqExposure.readoutPatternField.get();
    }

    public void setAcqPattern(NirSpecReadoutPattern iPattern) {
        acqExposure.readoutPatternField.set(iPattern);
    }

    public String getAcqMsaFile() {
        return acqMsaFile.get();
    }

    public void setAcqMsaFile(String iMsaFile) {
        acqMsaFile.set(iMsaFile);
    }

    public Set<JwstReferenceStar> getReferenceStars() {
        return acqRefStars.get();
    }

    public void setReferenceStars(Set<JwstReferenceStar> iRefStars) {
        acqRefStars.set(iRefStars);
    }

    public void addReferenceStar(String iRefStar) {
        try {
            acqRefStars.addValue(acqRefStars.findMatchInLegalValues(iRefStar));
        } catch (BrokenLinkException e) {
            // TODO: handle exception
        }
    }

    @Override
    public NirSpecSubarray getSubarray() {
        //default behavior, though others may override this
        return null;
    }
    //[end]

    //[start]--------------------Constraints--------------------
    @CosiConstraint
    @SuppressWarnings("unused")
    private void cosiSetLegalReferenceStarList() {
        if (getTinaDocument() != null) {
            JwstTargets lTargs = ((JwstProposalSpecification)getTinaDocument()).getTargets();
            acqRefStars.setLegalValues(lTargs.getChildren(JwstReferenceStar.class));
        }
    }
    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import java.util.List;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecSlit;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecSubarray;
import edu.stsci.jwst.apt.view.template.nirspec.NirSpecScienceTemplateFormBuilder;
import edu.stsci.tina.form.FormFactory;
import edu.stsci.tina.model.TinaColumnMultiField;
import edu.stsci.tina.model.TinaField;
import edu.stsci.tina.model.fields.CosiConstrainedSelection;
import edu.stsci.util.ArrayUtils;

/**
 * @author Rob Douglas
 *
 */
public class NirSpecFixedSlitTemplate extends NirSpecTargetAcqTemplate {
    //[start]--------------------Statics----------------------
    static {
        FormFactory.registerFormBuilder(NirSpecFixedSlitTemplate.class, new NirSpecScienceTemplateFormBuilder());
    }

    //[end]
    //[start]--------------------Fields-----------------------

    private final CosiConstrainedSelection<NirSpecSlit> slit = NirSpecTemplateFieldFactory.makeSlitField(this);
    private final CosiConstrainedSelection<NirSpecSubarray> subarray = NirSpecTemplateFieldFactory.makeSubarrayField(this);
    private final TinaColumnMultiField<NirSpecExposureSpecification> expTable = NirSpecTemplateFieldFactory.
makeNirSpecExposureTableField(this);
    {
        setProperties((TinaField[])ArrayUtils.addArrays(getProperties(),
                new TinaField[] {slit, subarray, expTable}));
    }

    //[end]
    //[start]--------------------Constructors-----------------
    public NirSpecFixedSlitTemplate () {
        super();
        Cosi.completeInitialization(this, NirSpecFixedSlitTemplate.class);
    }

    //[end]
    //[start]--------------------Accessors--------------------

    @Override
    public NirSpecSubarray getSubarray() {
        return subarray.get();
    }

    public void setSubarray(NirSpecSubarray iSubarray) {
        subarray.set(iSubarray);
    }

    public NirSpecSlit getSlit() {
        return slit.get();
    }

    public void setSlit(NirSpecSlit iSlit) {
        slit.set(iSlit);
    }

    public List<NirSpecExposureSpecification> getExposures() {
        return expTable.getValue();
    }
    //[end]
    //[start]--------------------Methods----------------------
    public void addExposure(NirSpecExposureSpecification<NirSpecFixedSlitTemplate> iRow) {
        expTable.addField(iRow);
    }
    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```

```
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import java.util.List;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.view.template.nirspec.NirSpecScienceTemplateFormBuilder;
import edu.stsci.tina.form.FormFactory;
import edu.stsci.tina.model.TinaColumnMultiField;
import edu.stsci.tina.model.TinaField;
import edu.stsci.util.ArrayUtils;

/**
 * @author Rob Douglas
 *
 */
public class NirSpecIfuTemplate extends NirSpecTargetAcqTemplate {
    //[start]--------------------Statics----------------------
    static {
        FormFactory.registerFormBuilder(NirSpecIfuTemplate.class, new NirSpecScienceTemplateFormBuilder());
    }

    //[end]
    //[start]--------------------Fields-----------------------
    private final TinaColumnMultiField<NirSpecExposureSpecification> expTable = NirSpecTemplateFieldFactory.
makeNirSpecExposureTableField(this);
    {
        setProperties((TinaField[])ArrayUtils.addArrays(getProperties(),
                new TinaField[] {expTable}));
        acqMsaFile.setRequired(false);
    }

    //[end]
    //[start]--------------------Constructors-----------------
    public NirSpecIfuTemplate () {
        super();
        Cosi.completeInitialization(this, NirSpecIfuTemplate.class);
    }

    //[end]
    //[start]--------------------Accessors--------------------


    //[end]
    //[start]--------------------Methods----------------------
    public List<NirSpecExposureSpecification> getExposures() {
        return expTable.getValue();
    }

    public void addExposure(NirSpecExposureSpecification<NirSpecIfuTemplate> iRow) {
        expTable.addField(iRow);
    }

    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import java.util.List;

import edu.stsci.CoSI.Cosi;
import edu.stsci.CoSI.CosiConstraint;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecReadoutPattern;
import edu.stsci.jwst.apt.view.template.nirspec.NirSpecMosTemplateFormBuilder;
import edu.stsci.tina.form.FormFactory;
import edu.stsci.tina.model.TinaColumnMultiField;
import edu.stsci.tina.model.TinaField;
import edu.stsci.tina.model.fields.CosiBooleanField;
import edu.stsci.tina.model.fields.TinaCosiStringField;
import edu.stsci.util.ArrayUtils;

/**
 * @author Rob Douglas
 *
 */
public class NirSpecMosTemplate extends NirSpecTargetAcqTemplate {
    //[start]--------------------Statics----------------------
    static {
        FormFactory.registerFormBuilder(NirSpecMosTemplate.class, new NirSpecMosTemplateFormBuilder());
    }

    //[end]
    //[start]--------------------Fields-----------------------
    private final CosiBooleanField irImageAvail = NirSpecTemplateFieldFactory.makeIRImageAvailField(this);
    private final TinaCosiStringField msaFile = NirSpecTemplateFieldFactory.makeMsaFileField(this);
    private final CosiBooleanField confirmImage = NirSpecTemplateFieldFactory.makeConfirmImageField(this);
    private final NirSpecMosConfirmationExpSpec confirmExp = new NirSpecMosConfirmationExpSpec(this);
    private final TinaColumnMultiField<NirSpecExposureSpecification> expTable = NirSpecTemplateFieldFactory.
makeNirSpecExposureTableField(this);
    {
        setProperties((TinaField[])ArrayUtils.addArrays(new TinaField[] {irImageAvail},
                getProperties()));
        setProperties((TinaField[])ArrayUtils.addArrays(getProperties(),
                new TinaField[] {confirmImage, confirmExp.readoutPatternField, confirmExp.numberOfGroupsField,
            confirmExp.numberOfIntegrationsField, msaFile, expTable}));
        acqMsaFile.setRequired(false);
    }

    //[end]
    //[start]--------------------Constructors-----------------
    public NirSpecMosTemplate () {
        super();
        add(confirmExp,false);
        Cosi.completeInitialization(this, NirSpecMosTemplate.class);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    public Boolean isIRImageAvailable() {
        return irImageAvail.get();
    }

    public void setIRImageAvailable(Boolean iAvailable) {
        irImageAvail.set(iAvailable);
    }

    public Boolean isOptionalConfirmImage() {
        return confirmImage.get();
    }

    public void setOptionalConfirmImage(Boolean iAvailable) {
        confirmImage.set(iAvailable);
    }

    public NirSpecMosConfirmationExpSpec getConfirmExposure () {
        return confirmExp;
    }

    public NirSpecReadoutPattern getConfirmationReadoutPattern() {
        return confirmExp.readoutPatternField.get();
    }

    public void setConfirmationReadoutPattern(NirSpecReadoutPattern iPatt) {
        confirmExp.readoutPatternField.set(iPatt);
    }

    public Integer getConfirmationGroups() {
        return confirmExp.numberOfGroupsField.get();
    }

    public void setConfirmationGroups(Integer iGroups) {
        confirmExp.numberOfGroupsField.set(iGroups);
    }

    public String getMsaFile() {
        return msaFile.get();
    }

    public void setMsaFile(String iFile) {
        msaFile.set(iFile);
    }

    public List<NirSpecExposureSpecification> getExposures() {
        return expTable.getValue();
    }
    //[end]
    //[start]--------------------Methods----------------------
    public void addExposure(NirSpecExposureSpecification<NirSpecMosTemplate> iRow) {
        expTable.addField(iRow);
    }

    //[end]
    //[start]--------------------Constraints------------------
    @CosiConstraint
    @SuppressWarnings("unused")
    private void cosiSetCofirmImageProperties() {
        for (TinaField lField : confirmExp.getProperties()) {
            lField.setEditable(confirmImage.get());
            lField.setRequired(confirmImage.get());
        }
    }
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import java.util.List;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecSubarray;
import edu.stsci.tina.model.TinaColumnMultiField;
import edu.stsci.tina.model.TinaField;

/**
 * @author Rob Douglas
 *
 */
public class NirSpecDarkTemplate extends NirSpecTemplate {
    //[start]--------------------Statics----------------------


    //[end]
    //[start]--------------------Fields-----------------------
    private final TinaColumnMultiField<NirSpecDarkExposureSpecification> expTable = NirSpecTemplateFieldFactory.
makeNirSpecDarkExposureTableField(this);
    {
        setProperties(new TinaField[] {expTable});
    }

    //[end]
    //[start]--------------------Constructors-----------------
    public NirSpecDarkTemplate () {
        super(true);
        Cosi.completeInitialization(this, NirSpecDarkTemplate.class);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    @Override
    public NirSpecSubarray getSubarray() {
        return null;
    }

    public List<NirSpecDarkExposureSpecification> getExposures() {
        return expTable.getValue();
    }

    //[end]
    //[start]--------------------Methods----------------------
    public void addExposure(NirSpecDarkExposureSpecification iRow) {
        expTable.addField(iRow);
    }

    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.*;
import edu.stsci.jwst.apt.model.template.JwstTemplateFieldFactory;
import edu.stsci.jwst.apt.view.template.nirspec.NirSpecInternalLampTemplateFormBuilder;
import edu.stsci.tina.form.FormFactory;
import edu.stsci.tina.model.TinaField;
import edu.stsci.tina.model.fields.CosiConstrainedInt;
import edu.stsci.tina.model.fields.CosiConstrainedSelection;
import edu.stsci.tina.model.fields.TinaCosiStringField;

/**
 * @author Rob Douglas
 *
 */
public class NirSpecInternalLampTemplate extends NirSpecTemplate {
    //[start]--------------------Statics----------------------
    static {
        FormFactory.registerFormBuilder(NirSpecInternalLampTemplate.class, new NirSpecInternalLampTemplateFormBuilder());
    }
    //[end]
    //[start]--------------------Fields-----------------------
    private final CosiConstrainedSelection<NirSpecOpMode> opmode = NirSpecTemplateFieldFactory.makeOpmodeField(this);
    private final TinaCosiStringField msaFile = NirSpecTemplateFieldFactory.makeMsaFileField(this);
    private final CosiConstrainedInt numExps  = JwstTemplateFieldFactory.makeNumberOfExpsField(this);
    private final NirSpecInternalLampExpSpec exposure = new NirSpecInternalLampExpSpec(this);
    private final CosiConstrainedSelection<NirSpecLamp> lamp = NirSpecTemplateFieldFactory.makeLampField(this);
    private final CosiConstrainedSelection<NirSpecGrating> grating = NirSpecTemplateFieldFactory.makeGratingField(this);
    {
        setProperties(new TinaField[] {opmode, msaFile, numExps,
                exposure.readoutPatternField, exposure.numberOfGroupsField,
                exposure.numberOfIntegrationsField, exposure.actualExpTimeField,
                lamp, grating});
    }


    //[end]
    //[start]--------------------Constructors-----------------
    public NirSpecInternalLampTemplate () {
        super(true);
        add(exposure,false);
        Cosi.completeInitialization(this, NirSpecInternalLampTemplate.class);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    @Override
    public NirSpecSubarray getSubarray() {
        return null;
    }

    public NirSpecOpMode getOpMode() {
        return opmode.get();
    }

    public void setOpMode(NirSpecOpMode iOpMode) {
        opmode.set(iOpMode);
    }

    public Integer getExposures() {
        return numExps.get();
    }

    public void setExposures(Integer iExp) {
        numExps.set(iExp);
    }

    public String getMsaFile() {
        return msaFile.get();
    }

    public void setMsaFile(String iFile) {
        msaFile.set(iFile);
    }

    public NirSpecInternalLampExpSpec getExposure() {
        return exposure;
    }

    public NirSpecReadoutPattern getPattern() {
        return exposure.getReadoutPattern();
    }

    public void setPattern(NirSpecReadoutPattern iPatt) {
        exposure.setReadoutPattern(iPatt);
    }

    public Integer getGroups() {
        return exposure.getNumberOfGroups();
    }

    public void setGroups(Integer iGroups) {
        exposure.setNumberOfGroups(iGroups);
    }

    public Integer getIntegrations() {
        return exposure.getNumberOfIntegrations();
    }

    public void setIntegrations(Integer iInts) {
        exposure.setNumberOfIntegrations(iInts);
    }

    public NirSpecLamp getLamp() {
        return lamp.get();
    }

    public void setLamp(NirSpecLamp iLamp) {
        lamp.set(iLamp);
    }

    public NirSpecGrating getGrating() {
        return grating.get();
    }

    public void setGrating(NirSpecGrating iGrating) {
        grating.set(iGrating);
    }



    //[end]
    //[start]--------------------Methods----------------------


    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------
     //[end]
}
```

```
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import java.util.List;

import edu.stsci.CoSI.Cosi;
import edu.stsci.CoSI.CosiConstraint;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecFilter;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecReadoutPattern;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecSubarray;
import edu.stsci.jwst.apt.view.template.nirspec.NirSpecFocusTemplateFormBuilder;
import edu.stsci.tina.form.FormFactory;
import edu.stsci.tina.model.TinaColumnMultiField;
import edu.stsci.tina.model.TinaField;
import edu.stsci.tina.model.fields.CosiConstrainedSelection;
import edu.stsci.tina.model.fields.TinaCosiStringField;
import edu.stsci.util.diagnostics.DiagnosticManager;

/**
 * @author Rob Douglas
 *
 */
public class NirSpecFocusTemplate extends NirSpecTemplate {
    //[start]--------------------Statics----------------------
    static {
        FormFactory.registerFormBuilder(NirSpecFocusTemplate.class, new NirSpecFocusTemplateFormBuilder());
    }
    //[end]
    //[start]--------------------Fields-----------------------
    private final TinaCosiStringField msaFile = NirSpecTemplateFieldFactory.makeMsaFileField(this);
    private final CosiConstrainedSelection<NirSpecFilter> filter = NirSpecTemplateFieldFactory.makeFilterField(this);
    private final NirSpecFocusExpSpec exposure = new NirSpecFocusExpSpec(this);
    private final TinaColumnMultiField<NirSpecFocusDelta> deltaArray = NirSpecTemplateFieldFactory.
makeNirSpecFocusDeltaTableField(this);
    {
        setProperties(new TinaField[] {msaFile, filter,
                exposure.readoutPatternField, exposure.numberOfGroupsField,
                exposure.numberOfIntegrationsField, exposure.actualExpTimeField,
                deltaArray});
    }

    //[end]
    //[start]--------------------Constructors-----------------
    public NirSpecFocusTemplate () {
        super();
        add(exposure,false);
        Cosi.completeInitialization(this, NirSpecFocusTemplate.class);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    @Override
    public NirSpecSubarray getSubarray() {
        return null;
    }

    public String getMsaFile() {
        return msaFile.get();
    }

    public void setMsaFile(String iFile) {
        msaFile.set(iFile);
    }

    public NirSpecFilter getFilter() {
        return filter.get();
    }
    public void setFilter(NirSpecFilter iFilter) {
        filter.set(iFilter);
    }

    public NirSpecFocusExpSpec getExposure() {
        return exposure;
    }

    public NirSpecReadoutPattern getReadoutPattern() {
        return exposure.getReadoutPattern();
    }

    public void setReadoutPattern(NirSpecReadoutPattern iPatt) {
        exposure.setReadoutPattern(iPatt);
    }

    public Integer getGroups() {
        return exposure.getNumberOfGroups();
    }

    public void setGroups(Integer iGroups) {
        exposure.setNumberOfGroups(iGroups);
    }

    public Integer getIntegrations() {
        return exposure.getNumberOfIntegrations();
    }

    public void setIntegrations(Integer iIntegrations) {
        exposure.setNumberOfIntegrations(iIntegrations);
    }

    public List<NirSpecFocusDelta> getRelativePositions() {
        return deltaArray.getValue();
    }

    public void addRelativePosition(NirSpecFocusDelta iDelta) {
        deltaArray.getValue().add(iDelta);
    }
    //[end]
    //[start]--------------------Methods----------------------


    //[end]
    //[start]--------------------Constraints------------------
    @CosiConstraint
    @SuppressWarnings("unused")
    private void cosiCheckLegalFocusPositions() {
        int lRows = deltaArray.getRowCount();
        DiagnosticManager.ensureDiagnostic(this, NirSpecTemplateFieldFactory.DELTA_ARRAY, deltaArray, DiagnosticManager.ERROR,
                "Must have 1 to 20 Positions",
                "Choose either 1 Position to move the Focus, or up to 20 to cycle through positions and return to start.",
                (lRows<1 || lRows>20));
    }

    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecDirection;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecPosition;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecSubarray;
import edu.stsci.tina.model.TinaField;
import edu.stsci.tina.model.fields.CosiConstrainedSelection;

/**
 * @author Rob Douglas
 *
 */
public class NirSpecFocusReferenceTemplate extends NirSpecTemplate {
    //[start]--------------------Statics----------------------


    //[end]
    //[start]--------------------Fields-----------------------
    CosiConstrainedSelection<NirSpecDirection> direction = NirSpecTemplateFieldFactory.makeDirectionField(this);
    CosiConstrainedSelection<NirSpecPosition> position = NirSpecTemplateFieldFactory.makePositionField(this);

    {
        setProperties(new TinaField[] {direction, position});
    }

    //[end]
    //[start]--------------------Constructors-----------------
    public NirSpecFocusReferenceTemplate () {
        super(true);
        Cosi.completeInitialization(this, NirSpecFocusReferenceTemplate.class);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    @Override
    public NirSpecSubarray getSubarray() {
        return null;
    }

    public NirSpecDirection getDirection() {
        return direction.get();
    }

    public void setDirection(NirSpecDirection iDir) {
        direction.set(iDir);
    }

    public NirSpecPosition getPosition() {
        return position.get();
    }

    public void setPosition(NirSpecPosition iPos) {
        position.set(iPos);
    }

    //[end]
    //[start]--------------------Methods----------------------


    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------
```

```
        //[end]
    }
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecSubarray;

/**
 * @author Rob Douglas
 *
 */
public class NirSpecMsaAnnealTemplate extends NirSpecTemplate {
    //[start]--------------------Statics----------------------


    //[end]
    //[start]--------------------Fields-----------------------


    //[end]
    //[start]--------------------Constructors-----------------
    public NirSpecMsaAnnealTemplate () {
        super(true);
        Cosi.completeInitialization(this, NirSpecMsaAnnealTemplate.class);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    @Override
    public NirSpecSubarray getSubarray() {
        return null;
    }

    //[end]
    //[start]--------------------Methods----------------------


    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;


import edu.stsci.CoSI.Calculator;
import edu.stsci.CoSI.Cosi;
import edu.stsci.CoSI.CosiConstraint;
import edu.stsci.CoSI.CosiDerivedProperty;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecGratingFilter;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecReadoutPattern;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecSubarray;
import edu.stsci.jwst.apt.model.template.JwstExposureSpecification;
import edu.stsci.jwst.apt.model.template.JwstTemplateFieldFactory;
import edu.stsci.jwst.apt.view.JwstExposureSpecificationFormBuilder;
import edu.stsci.tina.form.FormFactory;
import edu.stsci.tina.model.TinaField;
import edu.stsci.tina.model.fields.CosiConstrainedSelection;

/**
 * NirSpecExposureSpecification - configures the shared properties of all ways of expressing
 * exposure data in a NIRSPEC template, including access to the Derived parameter calculations.
 *
 * @author Rob Douglas
 */
public class NirSpecExposureSpecification<T extends NirSpecTemplate> extends JwstExposureSpecification {
    //--------------------Statics--------------------

    static {
        FormFactory.registerFormBuilder(NirSpecExposureSpecification.class,
                new JwstExposureSpecificationFormBuilder(JwstTemplateFieldFactory.REQUESTED_EXPOSURE_TIME));
    }

    //--------------------Fields--------------------
    protected final CosiConstrainedSelection<NirSpecGratingFilter> gratingFilter = NirSpecTemplateFieldFactory.
makeGratingFilterField(this);
    protected final CosiConstrainedSelection<NirSpecReadoutPattern> readoutPatternField = NirSpecTemplateFieldFactory.
makeReadoutPatternField(this);
    protected final T fTemplate;

    //--------------------Constructors--------------------
    public NirSpecExposureSpecification(T iTemplate) {
        super();
        fTemplate = iTemplate;
        actualExpTime = CosiDerivedProperty.createUninitializedProperty(this, 0.0, new CosiNirSpecActualExpTimeCalculator());
        initializeFields();
        configureEditableFields();
        Cosi.completeInitialization(this, NirSpecExposureSpecification.class);
    }

    //[start]--------------------Accessors--------------------
    public NirSpecSubarray getSubarray() {
        return fTemplate.getSubarray();
    }

    @Override
    public T getTemplate() {
        return fTemplate;
    }

    public NirSpecGratingFilter getGratingFilter() {
        return gratingFilter.get();
    }

    public Double getRequestedExpTime() {
        return requestedExpTime.get();
    }
    @Override
    public NirSpecReadoutPattern getReadoutPattern() {
        return readoutPatternField.get();
    }

    public void setGratingFilter(NirSpecGratingFilter iGratingFilter) {
        gratingFilter.set(iGratingFilter);
    }

    public void setRequestedExpTime(Double iRequestedExpTime) {
        requestedExpTime.set(iRequestedExpTime);
    }

    public void setRequestedExpTime(String iRequestedExpTime) {
        requestedExpTime.setValueFromString(iRequestedExpTime);
    }

    public void setReadoutPattern(NirSpecReadoutPattern iReadoutPattern) {
        readoutPatternField.set(iReadoutPattern);
    }

    //[end]

    //[start]--------------------Methods--------------------
    private void initializeFields() {
        setProperties(new TinaField[] {gratingFilter,
//                requestedExpTime,
                readoutPatternField,
                numberOfGroupsField,
                numberOfIntegrationsField,
                actualExpTimeField});
    }

    @Override
    protected void configureEditableFields() {
//        if (!JwstTemplate.isOpgsTestMode()) {
//            readoutPatternField.setEditable(false);
//            numberOfGroupsField.setEditable(false);
//            numberOfIntegrationsField.setEditable(false);
//        } else {
            requestedExpTime.setEditable(false);
//        }
    }

    //[start]-------------------- Constraints --------------------
    @CosiConstraint
    @SuppressWarnings("unused")
    private void cosiSetMaxGroups () {
        if (getReadoutPattern()!=null) {
            numberOfGroupsField.setMax(getReadoutPattern().getMaxGroups());
        }
    }
    //[end]

    //[start]-------------------- Inner Classes --------------------
    final class CosiNirSpecActualExpTimeCalculator implements Calculator<Double> {
        public Double calculate() {
            return 0.0;
        }
    }
    //[end]
}
```

```
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import java.util.Arrays;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecFilter;
import edu.stsci.jwst.apt.model.template.JwstTemplateFieldFactory;
import edu.stsci.jwst.apt.view.JwstExposureSpecificationFormBuilder;
import edu.stsci.tina.form.FormFactory;
import edu.stsci.tina.model.TinaField;
import edu.stsci.tina.model.fields.CosiConstrainedSelection;

/**
 * @author rob
 *
 */
public class NirSpecTargetAcqExposure extends NirSpecExposureSpecification<NirSpecTargetAcqTemplate> {
    public static final String ACQ_FILTER = "AcqFilter";
    static {
        FormFactory.registerFormBuilder(NirSpecTargetAcqExposure.class,
                new JwstExposureSpecificationFormBuilder(ACQ_FILTER,
                        JwstTemplateFieldFactory.REQUESTED_EXPOSURE_TIME));
    }

    protected final CosiConstrainedSelection<NirSpecFilter> acqFilter = NirSpecTemplateFieldFactory.makeFilterField(this,
ACQ_FILTER); {
        acqFilter.setLegalValues(Arrays.asList(NirSpecInstrument.ACQ_FILTERS));
        numberOfGroupsField.set(3);
        numberOfGroupsField.setEditable(false);
        numberOfIntegrationsField.set(1);
        numberOfIntegrationsField.setEditable(false);
    }

    //[start]-------------------- Constructors --------------------
    NirSpecTargetAcqExposure(NirSpecTargetAcqTemplate iTemplate) {
        super(iTemplate);
        setProperties(new TinaField[] {acqFilter, requestedExpTime, readoutPatternField,
                numberOfGroupsField, numberOfIntegrationsField,
                actualExpTimeField});
        Cosi.completeInitialization(this, NirSpecTargetAcqExposure.class);
    }
    //[end]
}
```

```
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.template.JwstTemplateFieldFactory;
import edu.stsci.jwst.apt.view.JwstExposureSpecificationFormBuilder;
import edu.stsci.tina.form.FormFactory;

/**
 * @author rob
 *
 */
public class NirSpecMosConfirmationExpSpec extends NirSpecExposureSpecification<NirSpecMosTemplate> {
    static {
        FormFactory.registerFormBuilder(NirSpecMosConfirmationExpSpec.class,
                new JwstExposureSpecificationFormBuilder(NirSpecTemplateFieldFactory.GRATING_FILTER,
                        JwstTemplateFieldFactory.REQUESTED_EXPOSURE_TIME));
    }

    //[start]-------------------- Constructors --------------------
    NirSpecMosConfirmationExpSpec(NirSpecMosTemplate iTemplate) {
        super(iTemplate);
        Cosi.completeInitialization(this, NirSpecMosConfirmationExpSpec.class);
    }
    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.instrument.NirSpecInstrument.NirSpecSubarray;
import edu.stsci.jwst.apt.model.template.JwstTemplateFieldFactory;
import edu.stsci.tina.model.fields.CosiConstrainedInt;

/**
 * NirSpecDarkExposureSpecTableRow - represents a single exposure specification for a Dark
 * image, including multiple exposures.
 *
 * @author Rob Douglas
 */
public class NirSpecDarkExposureSpecification extends NirSpecExposureSpecification<NirSpecDarkTemplate> {
    //--------------------Fields--------------------
    protected final CosiConstrainedInt numExps  = JwstTemplateFieldFactory.makeNumberOfExpsField(this);

    //--------------------Constructors--------------------
    public NirSpecDarkExposureSpecification (NirSpecDarkTemplate iContainer) {
        super(iContainer);
        gratingFilter.setEditable(false);
        gratingFilter.setRequired(false);
        requestedExpTime.setRequired(false);
        Cosi.completeInitialization(this,NirSpecDarkExposureSpecification.class);
    }

    //--------------------Accessors--------------------
    @Override
    public NirSpecSubarray getSubarray() {
        return null;
    }

    public void setNumberOfExposures (Integer iNumExps) {
        numExps.set(iNumExps);
    }

    public void setNumberOfExposures (String iNumExps) {
        numExps.setValueFromString(iNumExps);
    }

    public Integer getNumberOfExposures() {
        return numExps.get();
    }

    //--------------------Methods--------------------
    @Override
    protected void configureEditableFields() {
        //Don't change the legal editable fields.
    }
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.template.JwstTemplateFieldFactory;
import edu.stsci.jwst.apt.view.JwstExposureSpecificationFormBuilder;
import edu.stsci.tina.form.FormFactory;

/**
 * @author rob
 *
 */
public class NirSpecInternalLampExpSpec extends NirSpecExposureSpecification<NirSpecInternalLampTemplate> {
    static {
        FormFactory.registerFormBuilder(NirSpecInternalLampExpSpec.class,
                new JwstExposureSpecificationFormBuilder(NirSpecTemplateFieldFactory.GRATING_FILTER,
                        JwstTemplateFieldFactory.REQUESTED_EXPOSURE_TIME));
    }

    //[start]-------------------- Constructors --------------------
    NirSpecInternalLampExpSpec(NirSpecInternalLampTemplate iTemplate) {
        super(iTemplate);
        Cosi.completeInitialization(this, NirSpecInternalLampExpSpec.class);
    }
    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.template.JwstTemplateFieldFactory;
import edu.stsci.jwst.apt.view.JwstExposureSpecificationFormBuilder;
import edu.stsci.tina.form.FormFactory;

/**
 * @author rob
 *
 */
public class NirSpecFocusExpSpec extends NirSpecExposureSpecification<NirSpecFocusTemplate> {
    static {
        FormFactory.registerFormBuilder(NirSpecFocusExpSpec.class,
                new JwstExposureSpecificationFormBuilder(NirSpecTemplateFieldFactory.GRATING_FILTER,
                        JwstTemplateFieldFactory.REQUESTED_EXPOSURE_TIME));
    }

    //[start]-------------------- Constructors --------------------
    NirSpecFocusExpSpec(NirSpecFocusTemplate iTemplate) {
        super(iTemplate);
        Cosi.completeInitialization(this, NirSpecFocusExpSpec.class);
    }
    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.nirspec;

import org.jdom.Element;

import edu.stsci.jwst.apt.view.DefaultJwstFormBuilder;
import edu.stsci.tina.form.FormFactory;
import edu.stsci.tina.model.AbstractTinaDocumentElement;
import edu.stsci.tina.model.fields.CosiConstrainedInt;

/**
 * @author rob
 *
 */
public class NirSpecFocusDelta extends AbstractTinaDocumentElement {

    static {
        FormFactory.registerFormBuilder(NirSpecFocusDelta.class, new DefaultJwstFormBuilder());
    }

    //[start]--------------------Statics----------------------


    //[end]
    //[start]--------------------Fields-----------------------
    CosiConstrainedInt number = new CosiConstrainedInt(this, NirSpecTemplateFieldFactory.DELTA_NUMBER, true,
            NirSpecTemplateFieldFactory.DELTA_NUM_MIN,
            NirSpecTemplateFieldFactory.DELTA_NUM_MAX);
    CosiConstrainedInt delta = new CosiConstrainedInt(this, NirSpecTemplateFieldFactory.DELTA, true,
            NirSpecTemplateFieldFactory.DELTA_MIN,
            NirSpecTemplateFieldFactory.DELTA_MAX);
    //[end]
    //[start]--------------------Constructors-----------------
    public NirSpecFocusDelta(int iNum) {
        number.set(iNum);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    @Override
    public String getTypeName() {
        // TODO Auto-generated method stub
        throw new UnsupportedOperationException("This method has not been implemented.");
    }

    public Element getDomElement() {
        // TODO Auto-generated method stub
        throw new UnsupportedOperationException("This method has not been implemented.");
    }

    public Integer getDelta() {
        return delta.get();
    }

    public void setDelta(Integer iDelta) {
        delta.set(iDelta);
    }
    //[end]
    //[start]--------------------Methods----------------------


    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------
```

```
        //[end]
    }
```

```
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.tfi;

import java.util.ArrayList;
import java.util.List;

import edu.stsci.jwst.apt.model.instrument.TfiInstrument;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.*;
import edu.stsci.jwst.apt.model.template.JwstTemplate;
import edu.stsci.jwst.apt.model.template.JwstTemplateFieldFactory;
import edu.stsci.tina.model.*;
import edu.stsci.tina.model.fields.*;

/**
 * TfiTemplateFieldFactory - Provides a standard way to access the properties of a TFI
 * template that are shared among them.  Its only purpose is to return Fields that can be
 * used within the Templates.
 *
 * @author Rob Douglas
 */
public final class TfiTemplateFieldFactory extends JwstTemplateFieldFactory {
    //--------------------Statics--------------------
    public static final String SUBARRAY = "Subarray";
    public static final String FLAT_SUITE = "Flat Suite";
    public static final String READOUT_PATTERN = "Readout Pattern";
    public static final String EXPOSURES = "Exposures";
    public static final String FILTER = "Filter";
    public static final String WAVELENGTH = "Wavelength";
    public static final String MONO_IMAGE = "Monochromatic Image?";
    public static final String PUPIL = "Pupil";

    public static final String WAVE_RANGE = "Wavelength Range";
    public static final double MIN_WAVE = 1.500;
    public static final double MAX_WAVE = 5.000;

    public static final String DELTA_ARRAY = "Relative Positions";
    public static final String DELTA_NUMBER = "Number";
    public static final int DELTA_NUM_MIN = 1;
    public static final int DELTA_NUM_MAX = 20;
    public static final String DELTA = "Relative Position";
    public static final double DELTA_MIN = -9.0;
    public static final double DELTA_MAX = 3.6;

    public static final List<TableColumn<? super TfiExposureSpecification, ?>> TFI_EXP_COLUMNS = new ArrayList<TableColumn<?
super TfiExposureSpecification, ?>>();
    static {
        TFI_EXP_COLUMNS.add(new TfiWavelengthColumn());
        TFI_EXP_COLUMNS.add(new TfiFilterColumn());
        TFI_EXP_COLUMNS.add(new TfiMonoImageColumn());
        TFI_EXP_COLUMNS.add(new TfiReadOutPatternColumn());
        TFI_EXP_COLUMNS.add(new TfiNumOfGroupsColumn());
        TFI_EXP_COLUMNS.add(new TfiNumOfIntsColumn());
        TFI_EXP_COLUMNS.add(new TfiActualExpTimeColumn());
    }

    public static final RowFactory<TfiExposureSpecification> TFI_EXP_FACTORY = new RowFactory<TfiExposureSpecification>(){
        public TfiExposureSpecification<TfiTemplate> makeInstance(TinaColumnMultiField parent) {
            return new TfiExposureSpecification<TfiTemplate>((TfiTemplate)parent.getContainer());
        }
    };

    public static final List<TableColumn<? super TfiDarkExposureSpecification, ?>> TFI_DARK_COLUMNS = new ArrayList<
TableColumn<? super TfiDarkExposureSpecification, ?>>();
    static {
        TFI_DARK_COLUMNS.add(new TfiDarkNumberOfExpColumn());
        TFI_DARK_COLUMNS.add(new TfiReadOutPatternColumn());
        TFI_DARK_COLUMNS.add(new TfiNumOfGroupsColumn());
        TFI_DARK_COLUMNS.add(new TfiNumOfIntsColumn());
        TFI_DARK_COLUMNS.add(new TfiActualExpTimeColumn());
    }

    public static final RowFactory<TfiDarkExposureSpecification> TFI_DARK_FACTORY = new RowFactory<
TfiDarkExposureSpecification>(){
        public TfiDarkExposureSpecification makeInstance(TinaColumnMultiField parent) {
            return new TfiDarkExposureSpecification((TfiDarkTemplate)parent.getContainer());
        }
    };

    public static final List<TableColumn<? super TfiFocusDelta, ?>> TFI_FOCUS_DELTA_COLUMNS = new ArrayList<TableColumn<?
super TfiFocusDelta, ?>>();
    static {
        TFI_FOCUS_DELTA_COLUMNS.add(new TfiFocusDeltaNumberColumn());
        TFI_FOCUS_DELTA_COLUMNS.add(new TfiFocusDeltaColumn());
    }

    public static final RowFactory<TfiFocusDelta> TFI_FOCUS_DELTA_FACTORY = new RowFactory<TfiFocusDelta>(){
        public TfiFocusDelta makeInstance(TinaColumnMultiField parent) {
            return new TfiFocusDelta(parent.getRowCount()+1);
        }
    };

    //--------------------Constructors--------------------
    //Singleton - this is a static only class
    private TfiTemplateFieldFactory() {}

    //--------------------Methods--------------------
    public static CosiConstrainedSelection<TfiSubarray> makeSubarrayField (JwstTemplate<TfiInstrument> iIU) {
        return new CosiConstrainedSelectionBuilder<TfiSubarray>(iIU, SUBARRAY, true).setLegalValues(TfiSubarray.values()).
build();
    }

    public static CosiConstrainedSelection<TfiFlatSuite> makeFlatSuiteField (TfiInternalFlatTemplate iIU) {
        return new CosiConstrainedSelectionBuilder<TfiFlatSuite>(iIU, FLAT_SUITE, true).setLegalValues(TfiFlatSuite.values()).
build();
    }

    public static CosiConstrainedSelection<TfiReadoutPattern> makeReadoutPatternField (TinaDocumentElement iContainer) {
        return makeReadoutPatternField(iContainer, READOUT_PATTERN);
    }

    public static CosiConstrainedSelection<TfiReadoutPattern> makeReadoutPatternField (TinaDocumentElement iContainer, String
iName) {
        return new CosiConstrainedSelectionBuilder<TfiReadoutPattern>(iContainer, iName, true).setLegalValues
(TfiReadoutPattern.values()).build();
    }

    public static CosiBooleanField makeMonoImageField(TinaDocumentElement iContainer) {
        CosiBooleanField lMonoImage = new CosiBooleanField(iContainer, MONO_IMAGE, false);
//        lMonoImage.setTrueWord("Yes");
//        lMonoImage.setFalseWord("No");
        return lMonoImage;
    }

    public static CosiConstrainedSelection<TfiFilter> makeFilterField (TinaDocumentElement iIU) {
        return new CosiConstrainedSelectionBuilder<TfiFilter>(iIU, FILTER, true).setLegalValues(TfiFilter.values()).build();
    }

    public static CosiConstrainedDouble makeWavelengthField (TinaDocumentElement iIU) {
        return new CosiConstrainedDouble(iIU, WAVELENGTH, true, MIN_WAVE, MAX_WAVE);
    }

    public static CosiConstrainedSelection<TfiPupil> makePupilField (TfiFocusTemplate iIU) {
        return new CosiConstrainedSelectionBuilder<TfiPupil>(iIU, PUPIL, true).setLegalValues(TfiPupil.values()).build();
    }

    public static CosiConstrainedSelection<TfiWavelengthRange> makeWaveRangeField (TfiFocusTemplate iIU) {
        return new CosiConstrainedSelectionBuilder<TfiWavelengthRange>(iIU, WAVE_RANGE, true).setLegalValues
```

```
(TfiWavelengthRange.values()).build();
    }

    public static TinaColumnMultiField<TfiExposureSpecification> makeTfiExposureTableField (TfiTemplate iIU) {
        return new TinaColumnMultiField<TfiExposureSpecification>(iIU, EXPOSURES, null, TFI_EXP_COLUMNS, TFI_EXP_FACTORY);
    }

    public static TinaColumnMultiField<TfiDarkExposureSpecification> makeTfiDarkExposureTableField (TfiDarkTemplate iIU) {
        return new TinaColumnMultiField<TfiDarkExposureSpecification>(iIU, EXPOSURES, null, TFI_DARK_COLUMNS, TFI_DARK_FACTORY
);
    }

    public static TinaColumnMultiField<TfiFocusDelta> makeTfiFocusDeltaTableField (TfiFocusTemplate iIU) {
        return new TinaColumnMultiField<TfiFocusDelta>(iIU, DELTA_ARRAY, null, TFI_FOCUS_DELTA_COLUMNS,
TFI_FOCUS_DELTA_FACTORY);
    }

    //[start]--------------------Inner Classes--------------------
    //[start] Tfi Exposure Column Definitions (Template Independent)
    private static class TfiMonoImageColumn extends CosiBooleanColumn<TfiExposureSpecification>{
        public TfiMonoImageColumn() {
            super(TfiTemplateFieldFactory.MONO_IMAGE);
        }
        public CosiBooleanField get(TfiExposureSpecification getFrom) {
            return getFrom.monoImage;
        }
    }

    private static class TfiFilterColumn extends CosiConstrainedSelectionColumn<TfiExposureSpecification>{
        public TfiFilterColumn() {
            super(TfiTemplateFieldFactory.FILTER);
        }
        public CosiConstrainedSelection get(TfiExposureSpecification getFrom) {
            return getFrom.filter;
        }
    }

    private static class TfiWavelengthColumn extends CosiConstrainedDoubleColumn<TfiExposureSpecification>{
        public TfiWavelengthColumn() {
            super(TfiTemplateFieldFactory.WAVELENGTH);
        }
        public CosiConstrainedDouble get(TfiExposureSpecification getFrom) {
            return getFrom.wavelength;
        }
    }

    private static class TfiReqExpTimeColumn extends CosiConstrainedDoubleColumn<TfiExposureSpecification>{
        public TfiReqExpTimeColumn() {
            super(TfiTemplateFieldFactory.REQUESTED_EXPOSURE_TIME);
        }
        public CosiConstrainedDouble get(TfiExposureSpecification getFrom) {
            return getFrom.requestedExpTime;
        }
    }

    private static class TfiReadOutPatternColumn extends CosiConstrainedSelectionColumn<TfiExposureSpecification>{
        public TfiReadOutPatternColumn() {
            super(TfiTemplateFieldFactory.READOUT_PATTERN);
        }
        public CosiConstrainedSelection get(TfiExposureSpecification getFrom) {
            return getFrom.readoutPatternField;
        }
    }

    private static class TfiNumOfGroupsColumn extends CosiConstrainedIntColumn<TfiExposureSpecification>{
        public TfiNumOfGroupsColumn() {
            super(TfiTemplateFieldFactory.NUMBER_OF_GROUPS);
        }
        public CosiConstrainedInt get(TfiExposureSpecification getFrom) {
            return getFrom.numberOfGroupsField;
        }
    }

    private static class TfiNumOfIntsColumn extends CosiConstrainedIntColumn<TfiExposureSpecification>{
        public TfiNumOfIntsColumn() {
            super(TfiTemplateFieldFactory.NUMBER_OF_INTS);
        }
        public CosiConstrainedInt get(TfiExposureSpecification getFrom) {
            return getFrom.numberOfIntegrationsField;
        }
    }

    private static class TfiActualExpTimeColumn extends CosiConstrainedDoubleColumn<TfiExposureSpecification>{
        public TfiActualExpTimeColumn() {
            super(TfiTemplateFieldFactory.ACTUAL_EXP_TIME);
        }
        public CosiConstrainedDouble get(TfiExposureSpecification getFrom) {
            return getFrom.actualExpTimeField;
        }
    }

    //[end] Tfi Imaging Column Definitions

    //[start] Tfi Dark Column Definitions
    private static class TfiDarkNumberOfExpColumn extends CosiConstrainedIntColumn<TfiDarkExposureSpecification>{
        public TfiDarkNumberOfExpColumn() {
            super(TfiTemplateFieldFactory.NUMBER_OF_EXPS);
        }
        public CosiConstrainedInt get(TfiDarkExposureSpecification getFrom) {
            return getFrom.numExps;
        }
    }
    //[end] Tfi Dark Column Definitions

    //[start] Tfi Focus Delta Column Definitions
    private static class TfiFocusDeltaNumberColumn extends CosiConstrainedIntColumn<TfiFocusDelta>{
        public TfiFocusDeltaNumberColumn() {
            super(TfiTemplateFieldFactory.DELTA_NUMBER);
        }
        public CosiConstrainedInt get(TfiFocusDelta getFrom) {
            return getFrom.number;
        }
    }
    private static class TfiFocusDeltaColumn extends CosiConstrainedDoubleColumn<TfiFocusDelta>{
        public TfiFocusDeltaColumn() {
            super(TfiTemplateFieldFactory.DELTA);
        }
        public CosiConstrainedDouble get(TfiFocusDelta getFrom) {
            return getFrom.delta;
        }
    }
    //[end] NirSpec Focus Delta Column Definitions
    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.tfi;

import edu.stsci.jwst.apt.model.instrument.TfiInstrument;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.TfiSubarray;
import edu.stsci.jwst.apt.model.template.JwstTemplate;

/**
 * @author rob
 *
 */
public abstract class TfiTemplate extends JwstTemplate<TfiInstrument> {
    //[start]--------------------Statics----------------------


    //[end]
    //[start]--------------------Fields-----------------------


    //[end]
    //[start]--------------------Constructors-----------------
    public TfiTemplate() {
        this(false);
    }

    public TfiTemplate(boolean iInternal) {
        super(TfiInstrument.getInstance(), iInternal);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    public abstract TfiSubarray getSubarray();

    //[end]
    //[start]--------------------Methods----------------------


    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.tfi;

import java.util.List;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.TfiSubarray;
import edu.stsci.tina.model.TinaColumnMultiField;
import edu.stsci.tina.model.TinaField;
import edu.stsci.tina.model.fields.CosiConstrainedSelection;

/**
 * @author Rob Douglas
 *
 */
public class TfiImagingTemplate extends TfiTemplate {
    //[start]--------------------Statics----------------------


    //[end]
    //[start]--------------------Fields-----------------------
    private final CosiConstrainedSelection<TfiSubarray> subarray = TfiTemplateFieldFactory.makeSubarrayField(this);
    private final TinaColumnMultiField<TfiExposureSpecification> expTable = TfiTemplateFieldFactory.makeTfiExposureTableField
(this);
    {
        setProperties(new TinaField[] {subarray, expTable});
    }

    //[end]
    //[start]--------------------Constructors-----------------
    public TfiImagingTemplate () {
        super();
        Cosi.completeInitialization(this, TfiImagingTemplate.class);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    @Override
    public TfiSubarray getSubarray() {
        return subarray.get();
    }

    public void setSubarray (TfiSubarray iSubarray) {
        subarray.set(iSubarray);
    }

    public List<TfiExposureSpecification> getExposures() {
        return expTable.getValue();
    }

    //[end]
    //[start]--------------------Methods----------------------
    public void addExposure(TfiExposureSpecification<TfiImagingTemplate> iRow) {
        expTable.addField(iRow);
    }


    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.tfi;

import java.util.List;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.TfiSubarray;
import edu.stsci.tina.model.TinaColumnMultiField;
import edu.stsci.tina.model.TinaField;

/**
 * @author Rob Douglas
 *
 */
public class TfiDarkTemplate extends TfiTemplate {
    //[start]--------------------Statics----------------------


    //[end]
    //[start]--------------------Fields-----------------------
    private final TinaColumnMultiField<TfiDarkExposureSpecification> expTable = TfiTemplateFieldFactory.
makeTfiDarkExposureTableField(this);
    {
        setProperties(new TinaField[] {expTable});
    }

    //[end]
    //[start]--------------------Constructors-----------------
    public TfiDarkTemplate () {
        super(true);
        Cosi.completeInitialization(this, TfiDarkTemplate.class);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    @Override
    public TfiSubarray getSubarray() {
        return null;
    }

    public List<TfiDarkExposureSpecification> getExposures() {
        return expTable.getValue();
    }

    //[end]
    //[start]--------------------Methods----------------------
    public void addExposure(TfiDarkExposureSpecification iRow) {
        expTable.addField(iRow);
    }


    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.tfi;

import java.util.List;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.TfiPupil;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.TfiReadoutPattern;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.TfiSubarray;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.TfiWavelengthRange;
import edu.stsci.jwst.apt.view.template.tfi.TfiFocusTemplateFormBuilder;
import edu.stsci.tina.form.FormFactory;
import edu.stsci.tina.model.TinaColumnMultiField;
import edu.stsci.tina.model.TinaField;
import edu.stsci.tina.model.fields.CosiConstrainedSelection;

/**
 * @author Rob Douglas
 *
 */
public class TfiFocusTemplate extends TfiTemplate {
    //[start]--------------------Statics----------------------
    static {
        FormFactory.registerFormBuilder(TfiFocusTemplate.class, new TfiFocusTemplateFormBuilder());
    }

    //[end]
    //[start]--------------------Fields-----------------------
    private CosiConstrainedSelection<TfiPupil> pupil = TfiTemplateFieldFactory.makePupilField(this);
    private CosiConstrainedSelection<TfiWavelengthRange> waveRange = TfiTemplateFieldFactory.makeWaveRangeField(this);
    private TfiFocusExposureSpecification exposure = new TfiFocusExposureSpecification(this);
    private final TinaColumnMultiField<TfiFocusDelta> deltaArray = TfiTemplateFieldFactory.makeTfiFocusDeltaTableField(this);
    {
        setProperties(new TinaField[] {pupil, waveRange,
                exposure.readoutPatternField,
                exposure.numberOfGroupsField,
                exposure.numberOfIntegrationsField,
                exposure.actualExpTimeField,
                deltaArray});
    }

    //[end]
    //[start]--------------------Constructors-----------------
    public TfiFocusTemplate () {
        super();
        add(exposure,false);
        Cosi.completeInitialization(this, TfiFocusTemplate.class);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    @Override
    public TfiSubarray getSubarray() {
        return null;
    }

    public TfiExposureSpecification<TfiFocusTemplate> getExposure() {
        return exposure;
    }

    public TfiReadoutPattern getReadoutPattern () {
        return exposure.getReadoutPattern();
    }

    public void setReadoutPattern (TfiReadoutPattern iReadoutPattern) {
        exposure.setReadoutPattern(iReadoutPattern);
    }
    public Integer getNumberOfGroups () {
        return exposure.getNumberOfGroups();
    }

    public void setNumberOfGroups (Integer iNumGroups) {
        exposure.setNumberOfGroups(iNumGroups);
    }

    public void setNumberOfGroups (String iNumGroups) {
        exposure.setNumberOfGroups(iNumGroups);
    }

    public Integer getNumberOfIntegrations () {
        return exposure.getNumberOfIntegrations();
    }

    public void setNumberOfIntegrations (Integer iNumIntegrations) {
        exposure.setNumberOfIntegrations(iNumIntegrations);
    }

    public void setNumberOfIntegrations (String iNumIntegrations) {
        exposure.setNumberOfIntegrations(iNumIntegrations);
    }

    public TfiPupil getPupil () {
        return pupil.get();
    }

    public void setPupil (TfiPupil iPupil) {
        pupil.set(iPupil);
    }

    public TfiWavelengthRange getWaveRange () {
        return waveRange.get();
    }

    public void setWaveRange (TfiWavelengthRange iWaveRange) {
        waveRange.set(iWaveRange);
    }

    public List<TfiFocusDelta> getRelativePositions() {
        return deltaArray.getValue();
    }

    public void addRelativePosition(TfiFocusDelta iDelta) {
        deltaArray.getValue().add(iDelta);
    }

    //[end]
    //[start]--------------------Methods----------------------


    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.tfi;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.TfiFlatSuite;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.TfiSubarray;
import edu.stsci.tina.model.TinaField;
import edu.stsci.tina.model.fields.CosiConstrainedSelection;

/**
 * @author Rob Douglas
 *
 */
public class TfiInternalFlatTemplate extends TfiTemplate {
    //[start]--------------------Statics----------------------


    //[end]
    //[start]--------------------Fields-----------------------
    private final CosiConstrainedSelection<TfiFlatSuite> flatSuite = TfiTemplateFieldFactory.makeFlatSuiteField(this);

    {
        setProperties(new TinaField[] {flatSuite});
    }
    //[end]
    //[start]--------------------Constructors-----------------
    public TfiInternalFlatTemplate () {
        super(true);
        Cosi.completeInitialization(this, TfiInternalFlatTemplate.class);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    @Override
    public TfiSubarray getSubarray() {
        return null;
    }

    public TfiFlatSuite getFlatSuite () {
        return flatSuite.get();
    }

    public void setFlatSuite (TfiFlatSuite iFlatSuite) {
        flatSuite.set(iFlatSuite);
    }

    //[end]
    //[start]--------------------Methods----------------------


    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.tfi;


import edu.stsci.CoSI.Calculator;
import edu.stsci.CoSI.Cosi;
import edu.stsci.CoSI.CosiConstraint;
import edu.stsci.CoSI.CosiDerivedProperty;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.TfiFilter;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.TfiReadoutPattern;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.TfiSubarray;
import edu.stsci.jwst.apt.model.template.JwstExposureSpecification;
import edu.stsci.tina.model.TinaField;
import edu.stsci.tina.model.fields.CosiBooleanField;
import edu.stsci.tina.model.fields.CosiConstrainedDouble;
import edu.stsci.tina.model.fields.CosiConstrainedSelection;

/**
 * TfiExposureSpecification - configures the shared properties of all ways of expressing
 * exposure data in a TFI template, including access to the Derived parameter calculations.
 *
 * @author Rob Douglas
 */
public class TfiExposureSpecification<T extends TfiTemplate> extends JwstExposureSpecification {
    //--------------------Fields--------------------
    protected final CosiConstrainedDouble wavelength = TfiTemplateFieldFactory.makeWavelengthField(this);
    protected final CosiConstrainedSelection<TfiFilter> filter = TfiTemplateFieldFactory.makeFilterField(this); {
        filter.setEditable(false);
    }
    protected final CosiBooleanField monoImage = TfiTemplateFieldFactory.makeMonoImageField(this);
    protected final CosiConstrainedSelection<TfiReadoutPattern> readoutPatternField = TfiTemplateFieldFactory.
makeReadoutPatternField(this);
    protected final T fTemplate; {
        numberOfIntegrationsField.setMin(1);
        numberOfIntegrationsField.setMax(10);
    }

    //--------------------Constructors--------------------
    public TfiExposureSpecification(T iTemplate) {
        super();
        fTemplate = iTemplate;
        actualExpTime = CosiDerivedProperty.createUninitializedProperty(this, 0.0, new CosiTfiActualExpTimeCalculator());
        initializeFields();
        configureEditableFields();
        Cosi.completeInitialization(this, TfiExposureSpecification.class);
    }

    //[start]--------------------Accessors--------------------
    public TfiSubarray getSubarray() {
        return fTemplate.getSubarray();
    }

    @Override
    public T getTemplate() {
        return fTemplate;
    }

    public TfiFilter getFilter() {
        return filter.get();
    }

    public Double getRequestedExpTime() {
        return requestedExpTime.get();
    }

    @Override
    public TfiReadoutPattern getReadoutPattern() {
        return readoutPatternField.get();
    }

    public void setFilter(TfiFilter iFilter) {
        filter.set(iFilter);
    }

    public void setRequestedExpTime(Double iRequestedExpTime) {
        requestedExpTime.set(iRequestedExpTime);
    }

    public void setRequestedExpTime(String iRequestedExpTime) {
        requestedExpTime.setValueFromString(iRequestedExpTime);
    }

    public void setReadoutPattern(TfiReadoutPattern iReadoutPattern) {
        readoutPatternField.set(iReadoutPattern);
    }

    public Boolean isMonoImage () {
        return monoImage.get();
    }

    public void setMonoImage (Boolean iMonoImage) {
        monoImage.set(iMonoImage);
    }

    public String getWavelengthAsString() {
        return wavelength.getValueAsString();
    }

    public void setWavelengthFromString(String iValue) {
        wavelength.setValueFromString(iValue);
    }

    //[end]

    //[start]--------------------Methods--------------------
    private void initializeFields() {
        setProperties(new TinaField[] {wavelength, filter,
                monoImage,
//                requestedExpTime,
                readoutPatternField,
                numberOfGroupsField,
                numberOfIntegrationsField,
                actualExpTimeField});
    }

    @Override
    protected void configureEditableFields() {
//        if (!JwstTemplate.isOpgsTestMode()) {
//            readoutPatternField.setEditable(false);
//            numberOfGroupsField.setEditable(false);
//            numberOfIntegrationsField.setEditable(false);
//        } else {
            requestedExpTime.setEditable(false);
//        }
    }

    //[start]-------------------- Constraints --------------------
    @CosiConstraint
    @SuppressWarnings("unused")
    private void cosiSetMaxGroups () {
        if (getReadoutPattern()!=null) {
            numberOfGroupsField.setMax(getReadoutPattern().getMaxGroups());
        }
    }

    @CosiConstraint
    @SuppressWarnings("unused")
```

```
        private void cosiSetFilterFromWavelength() {
            if (wavelength.isSpecified()) {
                double lWave = wavelength.get();
                for (TfiFilter lFilter : TfiFilter.values()) {
                    if ((lWave >= lFilter.getMinWave()) &&
                            (lWave <= lFilter.getMaxWave())) {
                        filter.set(lFilter);
                        break;
                    }
                }
            }
        }
        //[end]

        //[start]-------------------- Inner Classes --------------------
        final class CosiTfiActualExpTimeCalculator implements Calculator<Double> {
            public Double calculate() {
                return 0.0;
            }
        }
        //[end]
    }
```

```
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.tfi;

import edu.stsci.CoSI.Cosi;
import edu.stsci.CoSI.CosiConstraint;
import edu.stsci.jwst.apt.model.instrument.TfiInstrument.TfiSubarray;
import edu.stsci.jwst.apt.model.template.JwstTemplateFieldFactory;
import edu.stsci.tina.model.fields.CosiConstrainedInt;

/**
 * TfiDarkExposureSpecTableRow - represents a single exposure specification for a Dark
 * image, including multiple exposures.
 *
 * @author Rob Douglas
 */
public class TfiDarkExposureSpecification extends TfiExposureSpecification<TfiDarkTemplate> {
    //[start]--------------------Fields--------------------
    protected final CosiConstrainedInt numExps  = JwstTemplateFieldFactory.makeNumberOfExpsField(this); {
        wavelength.setRequired(false);
        filter.setRequired(false);
        numberOfGroupsField.setEditable(false);
    }
    //[end]

    //[start]--------------------Constructors--------------------
    public TfiDarkExposureSpecification (TfiDarkTemplate iContainer) {
        super(iContainer);
        requestedExpTime.setRequired(false);
        Cosi.completeInitialization(this,TfiDarkExposureSpecification.class);
    }
    //[end]

    //[start]--------------------Accessors--------------------
    @Override
    public TfiSubarray getSubarray() {
        return null;
    }

    public void setNumberOfExposures (Integer iNumExps) {
        numExps.set(iNumExps);
    }

    public void setNumberOfExposures (String iNumExps) {
        numExps.setValueFromString(iNumExps);
    }

    public Integer getNumberOfExposures() {
        return numExps.get();
    }
    //[end]

    //[start]--------------------Methods--------------------
    @Override
    protected void configureEditableFields() {
        //Don't change the legal editable fields.
    }
    //[end]

    //[start]--------------------Constraints--------------------
    @CosiConstraint
    @SuppressWarnings("unused")
    private void cosiInferNumberOfGroups() {
        if (readoutPatternField.isSpecified()) {
            numberOfGroupsField.set(readoutPatternField.get().getMaxGroups());
        }
    }
    //[end]
    }
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.tfi;

import edu.stsci.CoSI.Cosi;
import edu.stsci.jwst.apt.view.JwstExposureSpecificationFormBuilder;
import edu.stsci.tina.form.FormFactory;

/**
 * @author rob
 *
 */
public class TfiFocusExposureSpecification extends TfiExposureSpecification<TfiFocusTemplate>  {
    //[start]--------------------Statics----------------------
    static {
        FormFactory.registerFormBuilder(TfiFocusExposureSpecification.class,
                new JwstExposureSpecificationFormBuilder(
                        TfiTemplateFieldFactory.MONO_IMAGE,
                        TfiTemplateFieldFactory.WAVELENGTH,
                        TfiTemplateFieldFactory.FILTER));
    }

    //[end]
    //[start]--------------------Fields-----------------------


    //[end]
    //[start]--------------------Constructors-----------------
    public TfiFocusExposureSpecification(TfiFocusTemplate iTemplate) {
        super(iTemplate);
        Cosi.completeInitialization(this, TfiFocusExposureSpecification.class);
    }

    //[end]
    //[start]--------------------Accessors--------------------


    //[end]
    //[start]--------------------Methods----------------------


    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```

```java
/**
 * This code was developed at the Space Telescope Science Institute under contract
 * to NASA.  All Rights Reserved.
 */
package edu.stsci.jwst.apt.model.template.tfi;

import org.jdom.Element;

import edu.stsci.jwst.apt.view.DefaultJwstFormBuilder;
import edu.stsci.tina.form.FormFactory;
import edu.stsci.tina.model.AbstractTinaDocumentElement;
import edu.stsci.tina.model.fields.CosiConstrainedDouble;
import edu.stsci.tina.model.fields.CosiConstrainedInt;

/**
 * @author rob
 *
 */
public class TfiFocusDelta extends AbstractTinaDocumentElement {

    static {
        FormFactory.registerFormBuilder(TfiFocusDelta.class, new DefaultJwstFormBuilder());
    }

    //[start]--------------------Statics----------------------


    //[end]
    //[start]--------------------Fields-----------------------
    CosiConstrainedInt number = new CosiConstrainedInt(this, TfiTemplateFieldFactory.DELTA_NUMBER, true,
            TfiTemplateFieldFactory.DELTA_NUM_MIN,
            TfiTemplateFieldFactory.DELTA_NUM_MAX);
    CosiConstrainedDouble delta = new CosiConstrainedDouble(this, TfiTemplateFieldFactory.DELTA, true,
            TfiTemplateFieldFactory.DELTA_MIN,
            TfiTemplateFieldFactory.DELTA_MAX);
    //[end]
    //[start]--------------------Constructors-----------------
    public TfiFocusDelta(int iNum) {
        number.set(iNum);
    }

    //[end]
    //[start]--------------------Accessors--------------------
    @Override
    public String getTypeName() {
        // TODO Auto-generated method stub
        throw new UnsupportedOperationException("This method has not been implemented.");
    }

    public Element getDomElement() {
        // TODO Auto-generated method stub
        throw new UnsupportedOperationException("This method has not been implemented.");
    }

    public Double getDelta() {
        return delta.get();
    }

    public void setDelta(Double iDelta) {
        delta.set(iDelta);
    }
    //[end]
    //[start]--------------------Methods----------------------


    //[end]
    //[start]--------------------Constraints------------------


    //[end]
    //[start]--------------------Inner Classes----------------


    //[end]
}
```