POWIEW Program Obliczeń Wielkich Wyzwań Nauki i Techniki

VisNow – tworzenie modułów i wtyczek

Szkolenie

16 stycznia 2014

Krzysztof S. Nowiński
Bartosz Borucki

visnow@icm.edu.pl

Interdyscyplinarne Centrum Modelowania Matematycznego i Komputerowego
Uniwersytet Warszawski

INNOWACYJNA GOSPODARKA
NARODOWA STRATEGIA SPÓJNOŚCI

icm

UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ ROZWOJU REGIONALNEGO

VisNow – moduły

- **Java**
  - VisNow jest stworzony w języku Java
  - Tworzenie modułów do VisNow'a – programowanie w Javie SE
  - Zalecany kod źródłowy w wersji Java 1.6

- **Moduły – podstawy**
  - Moduł = pakiet
  - Opis schematu modułu
  - Główna klasa modułu

- **Nazewnictwo pakietów**
  - Zgodne z zaleceniami nazewnictwa pakietów w Javie
  - **pl.moja.instytucja.projekt**
  - Np. pl.edu.icm.visnow, **pl.edu.uwm.visnow**
  - Moduły zwykle w podpakiecie **lib**, np: **pl.edu.uwm.visnow.lib**

INNOWACYJNA GOSPODARKA
NARODOWA STRATEGIA SPÓJNOŚCI
UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ
ROZWOJU REGIONALNEGO

VisNow – moduły

- **Opis schematu modułu – module.xml**
  - Plik XML module.xml w pakiecie
- **Definiuje**
  - Klasę główną (wymagane)
  - Nazwę (wymagane)
  - Skrócony opis
  - Porty wejściowe
  - Porty wyjściowe

Nazwa modułu widoczna w drzewie bibliotek i na przestrzeni roboczej w oknie głównym VisNow.

Opis modułu widoczny jako tooltip w drzewie bibliotek.

- **Struktura pliku**

```
<module name="my module name" class="ModuleMainClass">
    <inputs>
        Input definitions goes here
    </inputs>
    <outputs>
        Output definitions goes here
    </outputs>
    <description value="my module short description"/>
</module>
```

VisNow – moduły

- **Opis portu wejściowego – <input>**
  - Jeden wpis dla każdego portu

- **Definiuje**
  - Nazwę (wymagane)
  - Typ danych (wymagane)
    - pl.edu.icm.visnow.lib.types.VNField
    - pl.edu.icm.visnow.lib.types.VNRegularField
    - pl.edu.icm.visnow.lib.types.VNIrregularField
    - pl.edu.icm.visnow.lib.types.VNGeometryObject
    - java.lang.Float, Double, Byte, Short, Integer, Long, Boolean, Object
  - Modyfikatory (wymagane)
    - NECESSARY – port wymagany
    - TRIGGERING – port wyzwalający

Nazwa portu widoczna na module w przestrzeni roboczej i używana w kodzie klasy głównej modułu.

```
<input name="input_name" type="data_type_class" modifiers="port_modifiers">
        Acceptor definitions and description goes here
</input>
```

VisNow – moduły

- **Opis portu wyjściowego – <output>**
  - Jeden wpis dla każdego portu

- **Definiuje**
  - Nazwę (wymagane)
  - Typ danych (wymagane)
    - pl.edu.icm.visnow.lib.types.VNField
    - pl.edu.icm.visnow.lib.types.VNRegularField
    - pl.edu.icm.visnow.lib.types.VNIrregularField
    - pl.edu.icm.visnow.lib.types.VNGeometryObject
    - java.lang.Float, Double, Byte, Short, Integer, Long, Boolean, Object

Nazwa portu widoczna na module w przestrzeni roboczej i używana w kodzie klasy głównej modułu.

```
<output name="input_name" type="data_type_class">
        Schema definitions and description goes here
</output>
```

VisNow – moduły

- **Przykład module.xml**

```
<?xml version="1.0" encoding="UTF-8"?>
<module name="isosurface" class="Isosurface">
  <inputs>
    <input name="inField" type="pl.edu.icm.visnow.lib.types.VNField"
                                                    modifiers="NECESSARY:TRIGGERING">
      <description value="Input for data field to extract isosurface"/>
      <acceptor>
          <param name="NSPACE" value="3"/>
          <param name="REGULAR" value="true"/>
          <param name="NDIMS" value="3"/>
          <param name="DATA_VECLEN" value="1"/>
      </acceptor>
      <acceptor>
          <param name="NSPACE" value="3"/>
          <param name="IRREGULAR" value="true"/>
          <param name="DATA_VECLEN" value="1"/>
          <param name="CELLS_3D" value="true"/>
      </acceptor>
    </input>
```

- **Przykład module.xml**

```
    <input name="threshold" type="java.lang.Float" modifiers="TRIGGERING">
      <description value="Input for isolevel value"/>
    </input>
  </inputs>
  <outputs>
    <output name="isosurfaceField" type="pl.edu.icm.visnow.lib.types.VNIrregularField">
      <description value="Output for surface field"/>
      <schema>
        <param name="NSPACE" value="3"/>
        <param name="IRREGULAR" value="true"/>
        <param name="CELLS_TRIANGLE" value="true"/>
      </schema>
    </output>
    <geometryOutput/>
  </outputs>
  <description value="Maps volumetric data creating surface of constant data values."/>
</module>
```

VisNow – moduły

- **Główna klasa modułu**
  - Klasa której instancja jest tworzona w momencie utworzenia nowego modułu w przestrzeni roboczej
  - W ogólności dziedziczy po klasie **pl.edu.icm.visnow.engine.core.ModuleCore**
    - Komunikacja z silnikiem zarządzania przepływem
    - Obsługa portów
    - Obsługa zdarzeń
    - W praktyce – zbyt niskopoziomowy punkt wejścia
- **Visualization module**
  - Moduł zapewniający większość „administracji”
  - Daje domyślną prezentację wizualną danych wyjściowych
  - Dziedziczy po klasie **pl.edu.icm.visnow. … .OutFieldVisualizationModule**
- **Obowiązkowe elementy klasy głównej**
  - Nadpisana metoda **public void onActive() {}**
  - Pole **public static InputEgg[] inputEggs = null;**
  - Pole **public static OutputEgg[] inputEggs = null;**

Główna metoda obliczeniowa modułu. Wywoływana w momencie uruchomienia modułu.

VisNow – moduły

- **API modułu**
- **Metody do nadpisywania** (w ModuleCore lub OutFieldVisualizationModule)
  - **public void onActive()**
    - Wykonywana jako akcja modułu przy starcie (w osobnym wątku modułu)
  - **public void onInputAttach(LinkFace link)**
    - Wykonywana przy podłączeniu portu wejściowego
  - **public void onInputDetach(LinkFace link)**
    - Wykonywana przy odłączeniu portu wejściowego
  - **public void onOutputAttach(LinkFace link)**
    - Wykonywana przy podłączeniu portu wyjściowego
  - **public void onOutputDetach(LinkFace link)**
    - Wykonywana przy odłączeniu portu wyjściowego
  - **public void onDelete()**
    - Wykonywana po usunięciu modułu
  - **public boolean isGenerator()**
    - Powinna zwracać true jeśli moduł jest punktem wejścia do sieci modułów (np. odczyt danych lub generator danych) i zwykle nie posiada (obowiązkowych) portów wejściowych
  - **public boolean isViewer()**
    - Powinna zwracać true jeśli moduł jest końcowym modułem w sieci (np. Viewer) i nie posiada portów wyjściowych

- **API modułu**
- **Metody do wywoływania**
  - **String getName()**
    - Zwraca nazwę modułu.
  - **void startAction()**
    - Nieblokująca metoda używana do uruchomienia modułu. Powiadamia silnik sieci, że dany moduł chce się uruchomić. W konsekwencji wykonana zostanie metoda **onActive()** i moduły znajdujące się poniżej w sieci.
      UWAGA!! Bezpośrednie wywołanie metody **onActive()** nie powoduje uruchomienia modułu! Nie zostanie uruchomiona sieć i nie wykonają się moduły położone poniżej w sieci. Poza koniecznymi sytuacjami należy unikać wołania bezpośredniego metody **onActive()**!
  - **Vector<Object> getInputValues(String name)**
    - Metoda zwraca wektor wszystkich danych znajdujących się na porcie wejściowym. Parametr **"name"** to obowiązkowa nazwa portu z którego odczytujemy dane, zgodna z module.xml. Typ obiektów zwróconych będzie zgodny z typem danych portu (można je zrzutować).
  - **Object getInputFirstValue(String name)**
    - Metoda zwraca pierwszy obiekt danych (zwykle też jedyny) dostępny na porcie wejściowym. Parametr **"name"** to obowiązkowa nazwa portu z którego odczytujemy dane, zgodna z module.xml. Typ zwróconego obiektu będzie zgodny z typem danych portu (można go zrzutować).
  - **boolean setOutputValue(String name, Object value)**
    - Metoda ustawia dane na porcie wyjściowym. Parametr **"name"** to obowiązkowa nazwa portu wyjściowego na którym ustawiamy dane, zgodna z module.xml. Parametr **"value"** to obiekt który chcemy ustawić – musi być zgodny typem z typem danych portu.

- **API modułu**
- **Metody do wywoływania**
  - **boolean setProgress(float progress)**
    - Ustawia postęp wykonania modułu, reprezentowany prze pasek postępu na module. Ustawiana wartość powinna być z zakresu 0.0f-1.0f i odzwierciedlać postęp obliczeń. ***UWAGA!!*** Nie należy wołać tej metody zbyt często (np. w każdym przebiegu pętli) gdyż silnie obniży to wydajność.
  - **float getProgress()**
    - Zwraca aktualny postęp wykonania modułu, jeśli został ustawiony przez metodę setProgress(). Zwracana wartość jest z zakresu 0.0f-1.0f i reprezentuje postęp obliczeń.
  - **void setPanel(Component component)**
    - Ustawia komponentę Swing/AWT która reprezentuje interfejs użytkownika (GUI) modułu widoczny w panelu GUI modułu w oknie głównym VisNow'a, odpowiedzialny za sterowanie parametrami modułu. ***UWAGA!!*** Nie należy używać tej metody bezpośrednio w **OutFieldVisualizationModule**. Należy wówczas użyć polecenia **setPanel(ui)** w konstruktorze aby użyć domyślnego GUI prezentacji (zmienna **ui**) jako GUI modułu. Panel sterowania modułem może być dodany jako zakładka (computation tab) poprzez użycie metody **ui.addComputeGUI(JPanel gui)**.

- **API modułu – tylko w OutFieldVisualizationModule**
- **Pola/zmienne do wykorzystania**
  - **Field outfield;**
    - Należy używać tej zmiennej jako domyślnego pola wyjściowego modułu. Powoduje to automatyczną wizualizację tego pola (w domyślnej postaci, zgodnie z ustawieniami prezentacji na zakładce „presentation”, na domyślnym geometrycznym porcie wyjściowym) po wywołaniu metod **prepareOutputGeometry()** i **show()**.
  - **RegularField outRegularField;**
    - Należy używać tej zmiennej dla wygody obsługi pola regularnego. Zwykle używana do rzutowania pola outField.
  - **IrregularField outIrregularField;**
    - Należy używać tej zmiennej dla wygody obsługi pola nieregularnego. Zwykle używana do rzutowania pola outField.
- **Metody do wywoływania**
  - **void prepareOutputGeometry()**
    - Należy wywoływać tę metodę po stworzeniu pola wyjściowego i ustawieniu zmiennej outField. Tworzy domyślną reprezentację geometryczną pola outField. Zwykle wywoływana na końcu metody **onActive()**. Prawie zawsze następuje po niej wywołanie metody **show()**.
  - **void show()**
    - Należy wywoływać tę metodę po metodzie **prepareOutputGeometry()**. Dodaje stworzoną geometrię do obiektu wyjściowego reprezentowanego przez domyślny port geometryczny (<geometryOutput/>). Zwykle wywoływana na końcu metody **onActive()**. Prawie zawsze bezpośrednio po wywołaniu metody **prepareOutputGeometry ()**.

INNOWACYJNA GOSPODARKA
NARODOWA STRATEGIA SPÓJNOŚCI

icm

UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ
ROZWOJU REGIONALNEGO

VisNow – moduły

- **Następny krok**
  - **Podstawowy moduł to tylko module.xml i klasa modułu**
  - **W praktyce sporadyczne**
- **Pełna struktura modułu**
  - Opis schematu modułu
  - Główna klasa modułu
  - Klasa parametrów
  - Klasa interfejsu graficznego (GUI)
  - Klasa jądra obliczeniowego
    - Często lepiej wydzielić obliczenia z klasy głównej dla przejrzystości kodu
  - Dodatkowe klasy i zasoby
    - Wszystkie klasy i zasoby wykorzystywane tylko w danym module należy trzymać w pakiecie modułu lub podpakietach

INNOWACYJNA GOSPODARKA
NARODOWA STRATEGIA SPÓJNOŚCI
icm
UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ
ROZWOJU REGIONALNEGO

VisNow – moduły

- **Parametry**
  - **Klasa dziedzicząca po pl.edu.icm.visnow.engine.core.Parameters**
  - **Trzyma wewnętrzne parametry modułu, ustawiane z GUI, używane do obliczeń (np. wybrana komponenta, wartość progu, wymiar danych)**
  - **Odpowiada m.in. za zapis i odczyt parametrów przy zapisie/odczycie sieci (*)**
  - **W klasie głównej modułu istnieje zmienna „parameters” którą należy ustawić w konstruktorze.**
  - **Zwykle nazywana Params**
- **Definicja**
  - Statyczna tablica:

```
private static ParameterEgg[] eggs = new ParameterEgg[] {
        Parameter definitions goes here
};
```

- **Parametry**

  – **Każdy parametr definiowany jest przez element tej tablicy**

```
new ParameterEgg<param_class>("param_name",param_type,param_value)
```

  – *param_class* – klasa obiektu parametru (np. Float, Integer, int[])
  – *param_name* – nazwa parametru używana we wszystkich odwołaniach do tego parametru w setterach i getterach
  – *param_type* – jedna z trzech wartości: ParameterType.independent (parametr nie zależy od danych wejściowych modułu), ParameterType.dependent (parametr zależy od danych wejściowych modułu, np. wybrana komponenta) lub ParameterType.filename (ścieżka do pliku)
  – *param_value* – wartość domyślna parametru, obiekt klasy *param_value.* ***UWAGA!!*** Należy unikać ustawiania obiektów innych niż zmienne numeryczne w statycznym konstruktorze ze względu na współdzielenie obiektu pomiędzy kilka modułów tego samego typu. Należy wówczas ustawić **null** a wartość ustawić dopiero w konstruktorze klasy Params.

INNOWACYJNA GOSPODARKA
NARODOWA STRATEGIA SPÓJNOŚCI

icm

UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ
ROZWOJU REGIONALNEGO

VisNow – moduły

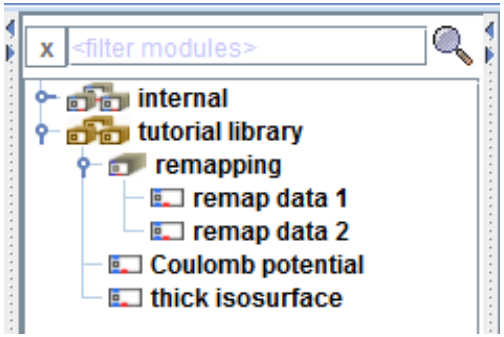
- **Parametry**
  - **Odwołanie do parametrów metodami:**
    - **void setValue(String name, Object value)** – aby ustawić wartość parametru “name”
    - **Object getValue(String name)** – aby pobrać aktualną wartość parametru “name”
  - **Dobra praktyka – napisać explicite setery i gettery dla każdego parametru**
- **Zdarzenia w parametrach**
  - **zdarzenie ParameterChange**
    - Wyzwalane za każdym wywołanie metody **setValue**
    - Używa interfejsu **pl.edu.icm.visnow.engine.core.ParameterChangeListener** do obsługi zdarzenia poprzez metodę **parameterChanged(String name)**.
    - Słuchacze mogą być dodawani metodą **addParameterChangelistener(ParameterChangeListener listener)**
    - Słuchacze mogą być usuwani metodą **removeParameterChangelistener(ParameterChangeListener listener)**
  - **zdarzenie StateChange**
    - Wyzwalane gdy zostanie wywołana metoda **fireStateChanged()**
    - Używa interfejsu **javax.swing.event.ChangeListener** do obsługi zdarzenia poprzez metodę **stateChanged(ChangeEvent e)**.
    - Słuchacze mogą być dodawani metodą **addChangeListener(ChangeListener listener)**
    - Słuchacze mogą być usuwani metodą **removeChangeListener(ChangeListener listener)**

VisNow – moduły

- **GUI**
  - **Komponent Swing/AWT**
  - **Zwykle klasa o nazwie GUI**
  - **Zwykle dziedzicząca po javax.swing.JPanel**
  - **Interfejs graficzny modułu – kontrolki sterowania parametrami modułu**
  - **Powinna być poprawnie ustawiana w konstruktorze**
  - **Widoczna w panelu modułu w oknie głównym VisNow**
  - **Zalecenie – widgety i panele predefiniowane w VisNow**
    - **pl.edu.icm.visnow.gui**
    - **pl.edu.icm.visnow.gui.widgets**
    - **pl.edu.icm.visnow.lib.gui**
  - ***UWAGA!!!***
    - Synchronizacja, unikanie zapętleń
    - Flagi blokujące lub flaga active parametrów

VisNow – wtyczki

- **Wtyczki (plugins)**
  - **Zewnętrzne biblioteki modułów**
  - **Nie modyfikujemy kodu VisNow**
  - **Doczytywane do aplikacji w run-time**
  - **Wtyczka = plik JAR**
    - **Moduły VisNow'a w pakietach**
    - **Pliki biblioteki**



- **Pliki bibliotek**
  - Pliki XML z opisem struktury drzewa biblioteki widocznej w panelu biblioteki
  - W głównym folderze pliku JAR
  - Zależne od wersji VisNow'a
    - **simple_library.xml** – dla VisNow Simple
    - **extended_library.xml** – dla VisNow Pro

```
<library name="my external library">
    <core package="your.institution.visnow.lib.module1"/>
    <core package="your.institution.visnow.lib.module2"/>
    <folder name="my folder">
        <core package="your.institution.visnow.lib.module3"/>
    </folder>
</library>
```


INNOWACYJNA GOSPODARKA
NARODOWA STRATEGIA SPÓJNOŚCI
icm
UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ
ROZWOJU REGIONALNEGO

VisNow – wtyczki

- **Tagi w library.xml**
  - **<library>** - tag korzeń definiujący bibliotekę:
    - **name** – parametr, nazwa biblioteki (widoczny w drzewie bibliotek), obowiązkowy
  - **<folder>** - tag grupowania modułów w podkatalogi/gałęzie drzewa bibliotek
    - **name** – parametr, nazwa folderu, obowiązkowy
    - **open** – parametr, ustawia czy folder jest domyślnie rozwinięty (wartość “yes” lub “no”), opcjonalny
    - **autosort** – parametr, ustawia czy zawartość folderu jest sortowana alfabetycznie (wartość “yes” lub “no”), opcjonalny
  - **<core>** - tag definiujący moduł
    - **package** – parametr, nazwa pakietu modułu, obowiązkowy
- **Typowa struktura wtyczki**