

# ArCADia - REINFORCED CONCRETE COMPONENT

---

User Manual

2019-03-07

# CONTENTS

1. EDITORS .....	4
1.1. Reinforced concrete component - basic information.....	5
1.2. Writing source code .....	5
2. GRAPHIC INTERFACE OF THE PROGRAM.....	6
2.1. Toolbar .....	7
2.2. Running the script in the ArCADia BIM system .....	9
2.3. The Script Explorer window .....	10
2.4. The Main Form.....	12
3. Defining the layout of the properties window .....	14
3.1. The structure of the document.....	15
3.2. The Image Panel.....	15
3.3. The Data Panel .....	17
3.4. The Row .....	18
3.5. The Column .....	19
3.6. The Pages .....	20
4. Defining controls .....	23
4.1. Using controls .....	24
4.2. The Editing Field.....	24
4.3. The Drop-down Lists .....	26
4.4. The radioGroup type button.....	29
4.5. The checkbox type button .....	30
5. Available features of the ArCADia BIM system .....	32
5.1. Dialogue and decision window .....	33
5.2. Inserting components.....	34
5.2.1. The empty 3D object .....	34
5.2.2. Adding a cuboid .....	35
5.2.3. Inserting a cuboid .....	37
5.2.4. Adding a cylinder .....	39
5.2.5. Adding an extruded solid.....	41
5.2.6. Adding the cutting of the solid to the concrete .....	42
5.2.7. Adding an extruded pyramid to the concrete solid.....	44
5.2.8. Adding an inverted extruded pyramid to a concrete solid.....	45
5.3. Inserting views .....	47

**EDITORS**

5.3.1.	The front view .....	47
5.3.2.	The rear view .....	48
5.3.3.	The view from the left .....	48
5.3.4.	The view from the right .....	49
5.4.	Inserting cross-sections .....	50
5.4.1.	The horizontal cross-section directed downwards .....	50
5.4.2.	Horizontal cross-section directed upwards .....	51
5.4.3.	The vertical cross-section directed to the right.....	51
5.4.4.	The vertical cross-section directed to the left.....	52
5.4.5.	The vertical cross-section directed to the front .....	53
5.4.6.	The vertical cross-section directed to the back.....	54
5.4.7.	Sample results of the script inserting selected cross-sections .....	55
5.5.	Inserting reinforcement.....	56
5.5.1.	The rebars .....	56
5.5.2.	The stirrups .....	58
5.6.	Inserting add-ons .....	59
5.6.1.	Szczegóły wszystkich prętów Details of all rebars .....	59
5.6.2.	Details of the rebars from the view - horizontally.....	61
5.6.3.	Details of the rebars from the view - vertically .....	62
5.6.4.	Missing details of the rebars .....	63
5.6.5.	A steel list of all elements.....	64
5.6.6.	A steel list of a single element.....	65
5.6.7.	The special characters in the text of controls .....	66
5.6.8.	Hatching.....	69
5.7.	The monitoring of script execution progress.....	81

# 1.EDITORS

## EDITORS

### 1.1. Reinforced concrete component - basic information

The RC component module is used to generate a three-dimensional 3D model of any RC component in the ArCADia BIM system (e.g. foundation footing, retaining wall, reinforced concrete beam, etc.), which was parameterized in a text file using the Lisp script language and the ArCADia BIM system functions. Based on the created 3D model of a reinforced concrete component, drawing documentation can be created, i.e. automatically create views with cross-sections, insert bar details, describe the reinforcement distribution, insert reinforcement steel list, etc. The system user can create the source code of the scripts himself (based on the shared documentation) and generate reinforced concrete components of any shape and reinforcement distribution. The user can also use the ready-made script solutions attached to the program and generate a 3D model of a reinforced concrete component along with the drawing documentation. The source code of the scripts is open, so the user can edit the code of the scripts included in the program or create own scripts.

### 1.2. Writing source code

Lisp scripts for the ArCADia BIM system can be written in any text editor, in which the appropriate commands known from the Lisp script language are entered and the corresponding functions available in the ArCADia BIM system are called. To facilitate the writing of the source code, it is recommended to use specialized text editors that enable syntax coloring for the selected programming language. A specialized editor after correctly entering a keyword for a given programming language automatically distinguishes it with a different color. Syntax coloring allows for faster writing of the source code of the script and easier searching for errors. Due to the fact that the scripts for the ArCADia BIM system contain a combination of two languages, i.e. Lisp and XML, it is recommended to choose a text editor that will support both programming languages. One such editor that meets the requirements is the Notepad ++ text editor.

## 2. GRAPHIC INTERFACE OF THE PROGRAM

## GRAPHIC INTERFACE OF THE PROGRAM

## 2.1. Toolbar

The Toolbar (Fig. 1) allows access to all the functions of the **Reinforced concrete component** module.

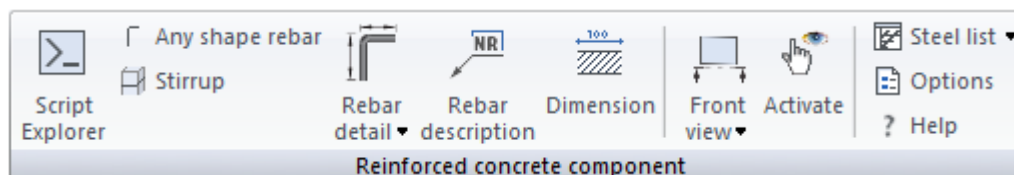


Fig. 1 The toolbar of the Reinforced concrete component module

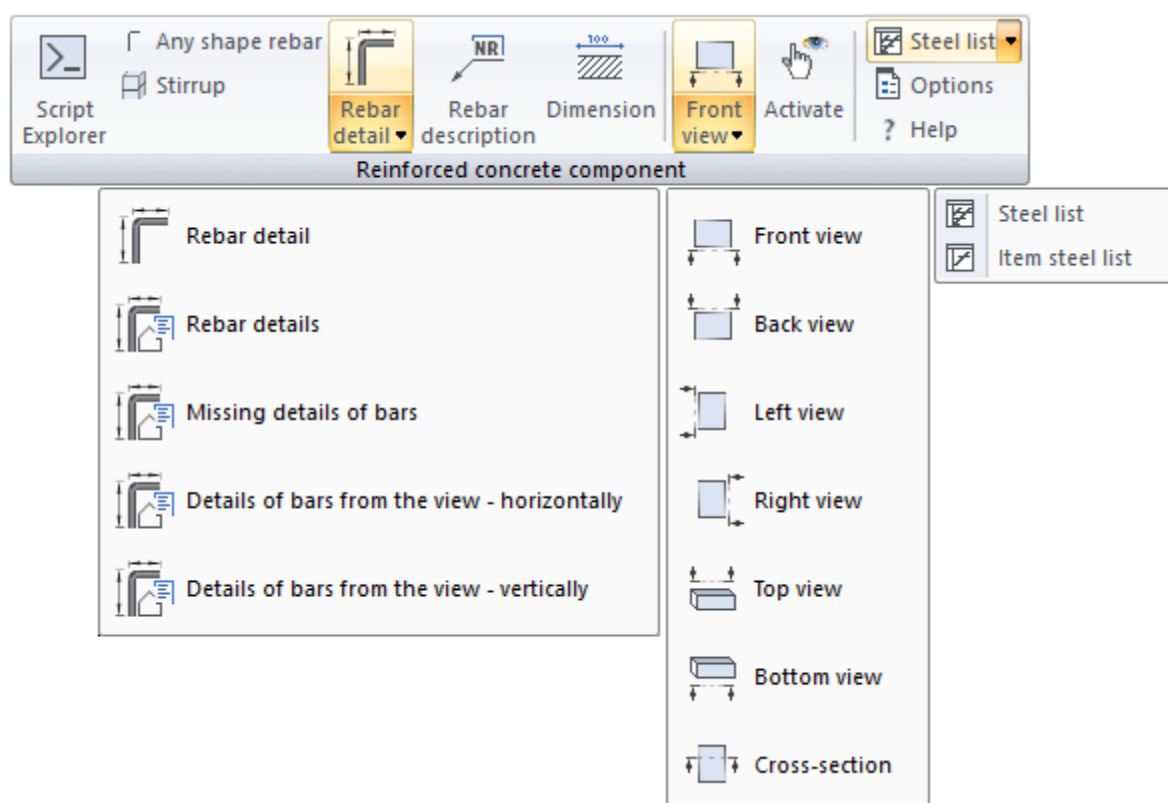











Fig. 2 The reinforced concrete component bar with expanded options

**\*BIM** – the options available to ArCADia BIM license holders, i.e. after purchasing one of the following programs: ArCADia, ArCADia AC, ArCADia LT or ArCADia PLUS.

Tab. 1. The functions of the ArCADia-REINFORCED CONCRETE COMPONENT module:

Icon	Option	Description	*BIM
	<i>Script Explorer</i>	Runs the <i>Script Explorer</i> window.	<b>X</b>

## GRAPHIC INTERFACE OF THE PROGRAM

	<i>Any shape rebar</i>	Allows inserting any reinforcing bar into the reinforced concrete model. The shape of the bar is defined on the active view/cross-section.	<b>X</b>
	<i>Stirrup</i>	Allows to insert stirrup/stirrup set into the the reinforced concrete model on the active view/cross-section.	<b>X</b>
	<i>Rebar detail</i>	Generates detail for the selected rebar/selected set of rebars.	<b>X</b>
	<i>Rebar details</i>	Generates details for all rebars instered in reinforced object.	<b>X</b>
	<i>Missing rebar details</i>	Generates missing details for all rebars instered in the project.	<b>X</b>
	<i>Rebar details from the view - horizontally</i>	Generates details of rebars visible on a given view/cross-section and distribute them horizontally.	<b>X</b>
	<i>Rebar details from the view - vertically</i>	Generates details of rebars visible on a given view/cross-section and distribute them vertically.	<b>X</b>
	<i>Rebar description</i>	Inserts a description for the selected rebar/selected rebars.	<b>X</b>
	<i>Dimension</i>	Inserts any dimension on the view/ cross-section	<b>X</b>
	<i>Steel list</i>	Inserts steel lists of all reinforced concrete elements currently in the project, taking into account the number of pieces given in the properties of reinforced concrete elements.	<b>X</b>
	<i>Element steel list</i>	Inserts a rebar List for the one element of reinforced concrete.	<b>X</b>
	<i>Front View</i>	Inserts the front view for a reinforced concrete component.	<b>X</b>
	<i>Back View</i>	Inserts the back view for a reinforced concrete component.	<b>X</b>
	<i>Left View</i>	Inserts the left view for a reinforced concrete component.	<b>X</b>
	<i>Right View</i>	Inserts the right view for a reinforced concrete component.	<b>X</b>
	<i>Top View</i>	Inserts the top view for a reinforced concrete component.	<b>X</b>
	<i>Bottom View</i>	Inserts the bottom view for a reinforced concrete component.	<b>X</b>
	<i>Cross-section</i>	Starts inserting a cross-section (horizontal or vertical) at any point in the active view.	<b>X</b>
	<i>Activate</i>	Activates the view/cross-section.	<b>X</b>
	<i>Options</i>	Activates the project options window	<b>X</b>
	<i>Help</i>	Opens the module help window	<b>X</b>



## GRAPHIC INTERFACE OF THE PROGRAM

## 2.2. Running the script in the ArCADia BIM system

The Lisp script of the reinforced concrete component is loaded by clicking the **Script Explorer** button (Fig. 3), which is located on the **reinforced concrete component** bar on the **Construction ribbon**.

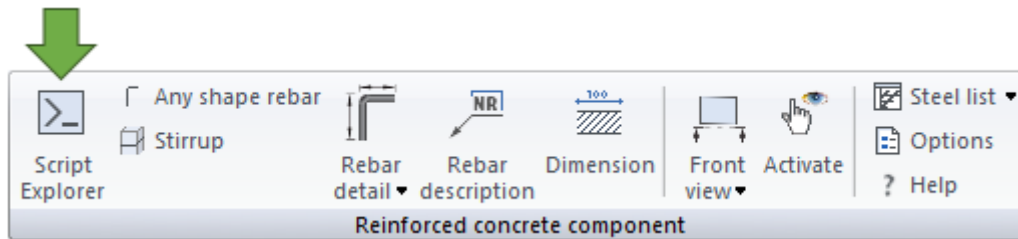


Fig. 3 The script explorer in the reinforced concrete component toolbar

In the left part of the **Script Explorer** window, we first select a group of scripts, and then in the right part of the window, we select the appropriate reinforced concrete component script. After selecting the element on the window, we click **Run**. The selection of the element is shown in Fig. 4.

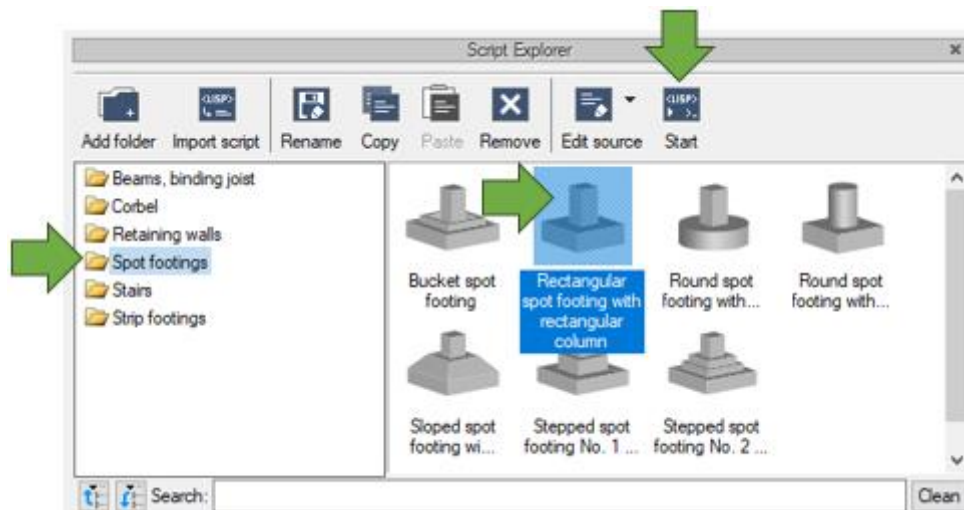


Fig. 4 The selection of the spot footing script in the Script Explorer window

If the source code of the loaded script has been correctly defined, the reinforced concrete component **Properties** window will be displayed (Fig. 5). Otherwise, an error message will be displayed.

## GRAPHIC INTERFACE OF THE PROGRAM

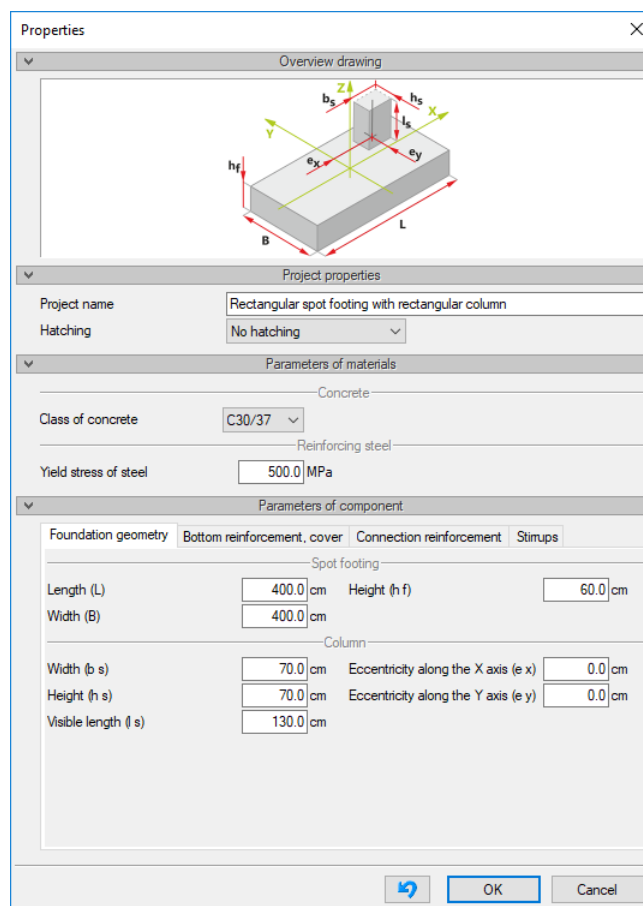


Fig. 5 A sample properties window of the reinforced concrete component loaded from the Lisp script

## 2.3. The Script Explorer window

The **Script Explorer** window (Fig. 6) gives the ability of quick and comfortable reinforced concrete components script management. In the window, among others you can add folders that are used to group imported or copied scripts. Using the options in the window, you can directly enable editing the source code of the script or display the properties window of the selected reinforced concrete element.

## GRAPHIC INTERFACE OF THE PROGRAM

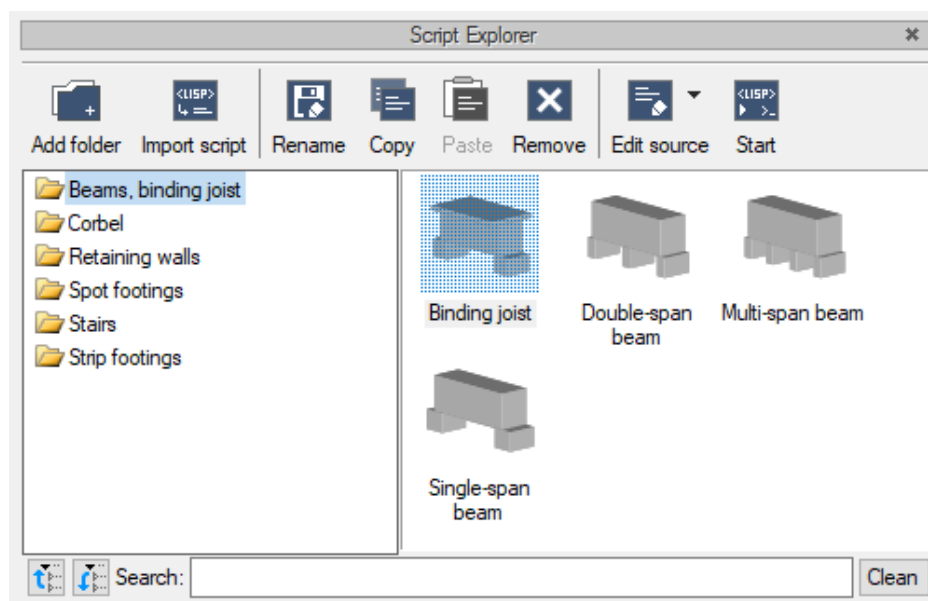








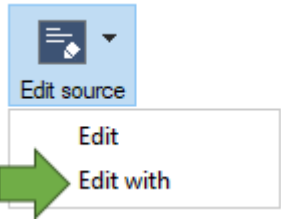



Fig. 6 The script Explorer window

Tab. 2 The functions of the Script Explorer

Ikona	Opcja	Opis
 Add folder	<i>Add folder</i>	Adds a folder (in the left part of the Script Explorer window), which will be used to group the reinforced concrete components scripts.
 Import script	<i>Import Script</i>	Imports the indicated Lisp script into the script database in the program and displays it in the previously selected folder in the Script Explorer window.
 Rename	<i>Rename</i>	Changes the name of the file/folder in the Script Explorer window.
 Copy	<i>Copy</i>	Copies the file/folder in the Script Explorer window.
 Paste	<i>Paste</i>	Pastes the file/folder in the Script Explorer window.
 Remove	<i>Remove</i>	Removes the file/folder in the Script Explorer window.
 Edit source  Edit Edit with	<i>Edit source – Edit</i>	Opens the source code of the Lisp script file of the selected reinforced concrete component in the associated editing program.

## GRAPHIC INTERFACE OF THE PROGRAM

	<i>Edit source – Edit with</i>	Indication of the program in which the Lisp file of the reinforced concrete component script selected for editing should be running.
	<i>Start</i>	Opens the Properties window of the selected reinforced concrete component.

The image of the icon displayed in the **Script Explorer** window for a given script should have the same name as the script and be in the same folder as the script.

## 2.4. The Main Form

After loading the script in ArCADia program, the **Properties** window will be displayed. The elements displayed in the window are defined directly in the source code contained in the Lisp file. On the basis of the source code of the script, the components (data panels, image panel, tabs, etc.) and controls (edit fields, drop-down lists, check boxes) are created. If the basic script code for the ArCADia BIM system is defined in the Lisp file, the basic **Properties** window will appear after running the script.

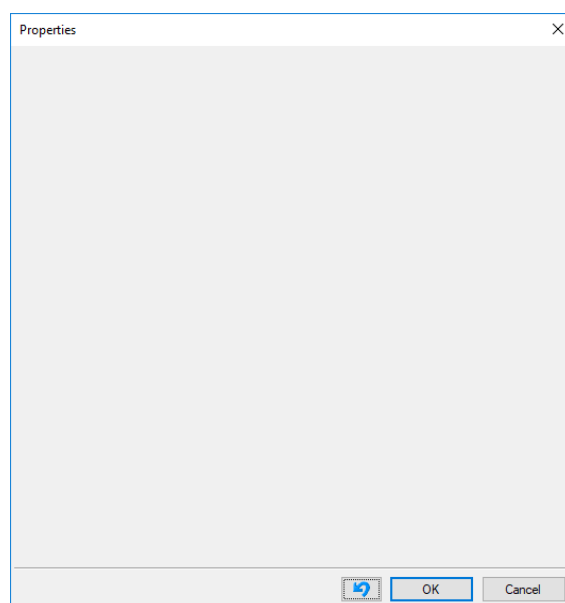

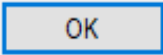
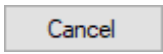


Fig. 7 The basic Properties window

In the lower part of the window there are buttons that support the appropriate form saving.

## GRAPHIC INTERFACE OF THE PROGRAM

Tab. 3 The functions of the Properties window

Ikona	Opcja	Opis
	<i>Reset initial settings</i>	Restores the default values for the controls defined directly in the Lisp file.
	<i>OK</i>	Saves the current data set in the controls to the loaded Lisp file and starts the execution of the script.
	<i>Cancel</i>	Discontinues the operations performed on the form and closes the window without saving the entered data.

By defining the appropriate components and controls in the Lisp file, you can design the appearance of the properties window. The main component in the window structure is the panel. Inside the panels, the rows are defined directly. In addition, you can create bookmarks inside the rows (the tabs inside must contain embedded lines). The rows are further divided into columns. The last step is defining the controls, which are arranged inside the columns. A more detailed description of window creation is presented in the further part of the document.

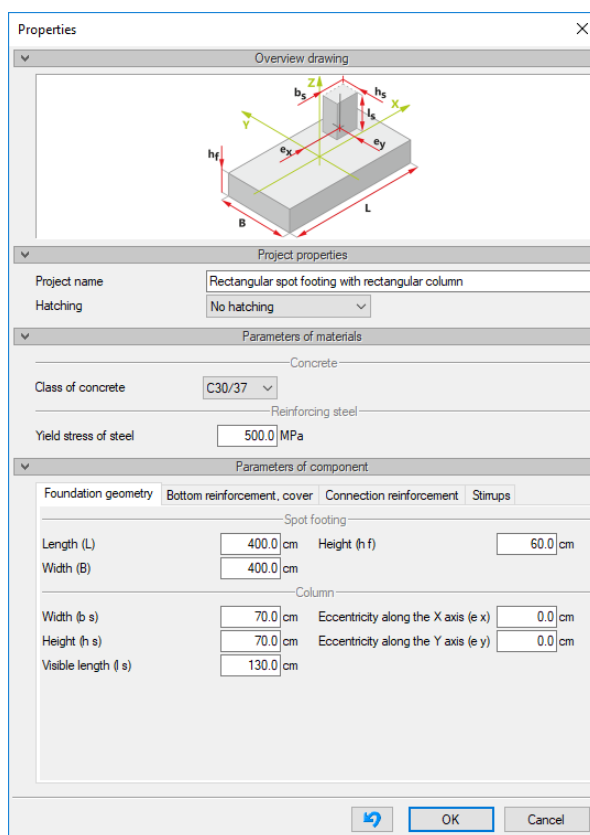


Fig. 8 A sample Properties window with components and controls

### 3. DEFINING THE LAYOUT OF THE PROPERTIES WINDOW

## Defining the layout of the properties window

### 3.1. The structure of the document

The Lisp script for the ArCADia BIM system is created as a regular text document. After running the text editor, a new file should be created and then saved with the extension \*.lsp or \*.lisp.

Each script should contain the basic source code, which is presented later on in the document. This code is a template of a typical Lisp script for the ArCADia BIM system.

The source code of the basic Lisp file for the ArCADia BIM system:

```

;#xml#<dialog>
;#xml# <panels>
;#xml# </panels>
;#xml#</dialog>
;#lispParams#
;Here, the list of lisp variables should be entered
;#lispParams#

;The executable code of the script is being entered below

```

Three most important parts can be distinguished in the document structure. The user's graphical interface which is created between the tags, ;#xml#<dialog> and ;#xml#</dialog>. Between the tags, #lispParams#; #lispParams# the typical variables used in the Lisp language are defined. In the further part of the script, the script code in the Lisp language is defined.

### 3.2. The Image Panel

In the program window, the so-called image panels can be created. This is a panel in which the image of the file is displayed, whose name along with the extension is given between the <image> </image>. tags. The displayed image must be in the same folder as the Lisp script file.

#### Properties

Name	Description
Id	The Id values
Name	The text on the ribbon
defaultFoldState	<p>The state of the panel after the content of the script has been loaded.</p> <p>Values:</p> <ul style="list-style-type: none"> <li>0 –dropped-down</li> <li>1 –rolled-up</li> </ul>

## Defining the layout of the properties window

Image	The name and extension of the displayed image (the image must be in the same folder as the loaded script)
-------	---

**Example**

The source code of the script with one image panel:

```
;#xml#<dialog>
;#xml# <panels>
;#xml#     <panel id="0">
;#xml#         <name>Overview drawing</name>
;#xml#         <defaultFoldState>0</defaultFoldState>
;#xml#         <image>imageName.png</image>
;#xml#     </panel>
;#xml# </panels>
;#xml#</dialog>
;#lispParams#
;#lispParams#
```

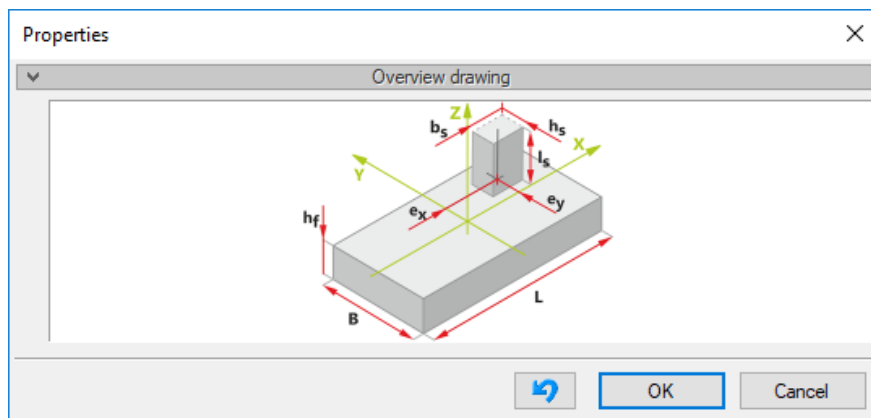


Fig. 9 The image panel - defaultFoldState value is equal 0 after the script was loaded

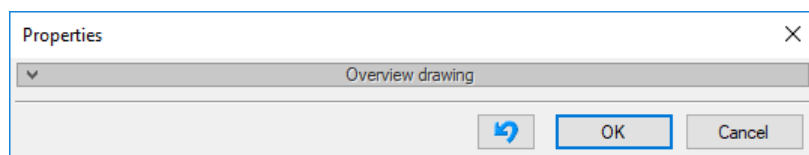


Fig. 10 The image panel - defaultFoldState value is equal 1 after the script was loaded



## Defining the layout of the properties window

### 3.3. The Data Panel

The data panels can be created in the program window. The panels are components in which tabs and controls are placed. All elements in the panel are organized by using rows and columns. These panel elements are defined between the `<rows>` `</rows>` tags.

**Panel tag properties**

Name	Description
Id	id values
Name	The text in the top bar
defaultFoldState	The state of the panel after the content of the script has been loaded. Values: <ul style="list-style-type: none"> <li>• 0 –dropped-down</li> <li>• 1 –rolled-up</li> </ul>
Rows	Row marker

**Example**

The source code of the script section with the data panel definition:

```

;#xml#      <panel id="0">
;#xml#      <name>Parameters of materials</name>
;#xml#      <defaultFoldState>0</defaultFoldState>
;#xml#      <rows>
;A panel content (rows, tabs, columns, controls)
;#xml#      </rows>
;#xml#      </panel>

```

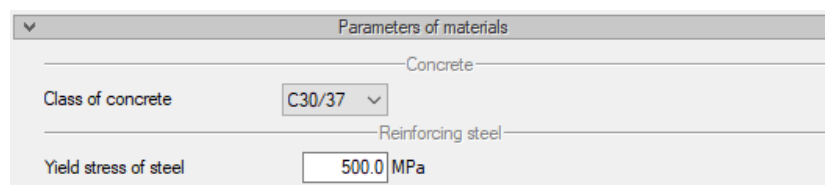


Fig. 11 A sample data panel - the defaultFoldState value is equal to 0 after loading the script

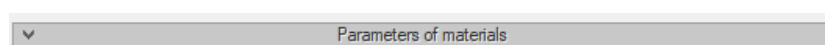


Fig. 12 A sample data panel - the defaultFoldState value is equal to 1 after loading the script

## Defining the layout of the properties window

### 3.4. The Row

A subordinate element of the panel are the rows that allow us to organize the controls by topic. The rows are subdivided into tabs or directly into columns.

**Row tag properties**

Name	Description
Id	id values
name	The text in the middle of the line symbolizing the beginning of the row
columns	Column tag
pages	Page tag

**Example**

The source code of the script fragment with the data panel definition of a single row divided further into columns:

```

;#xml#           <rows>
;#xml#           <row id="0">
;#xml#           <name>Concrete</name>
;#xml#           <columns>
; the definition of columns with controls
;#xml#           </columns>
;#xml#           </row>
;#xml#           </rows>

```

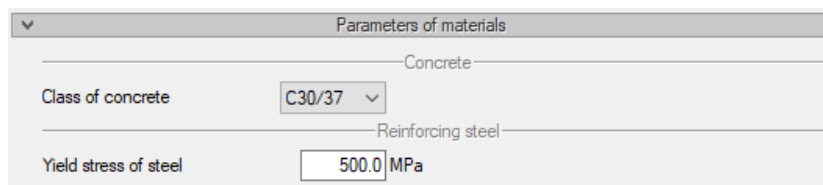


Fig. 13 A sample panel with data divided into two rows

A fragment of the source code of the script with the definition of a single row divided in a further part into tabs:

## Defining the layout of the properties window

```

;#xml#           <rows>
;#xml#           <row id="0">
;#xml#           <pages>
;tab, rows, columns and parameters definition
;#xml#           </pages>
;#xml#           </row>
;#xml#           </rows>

```

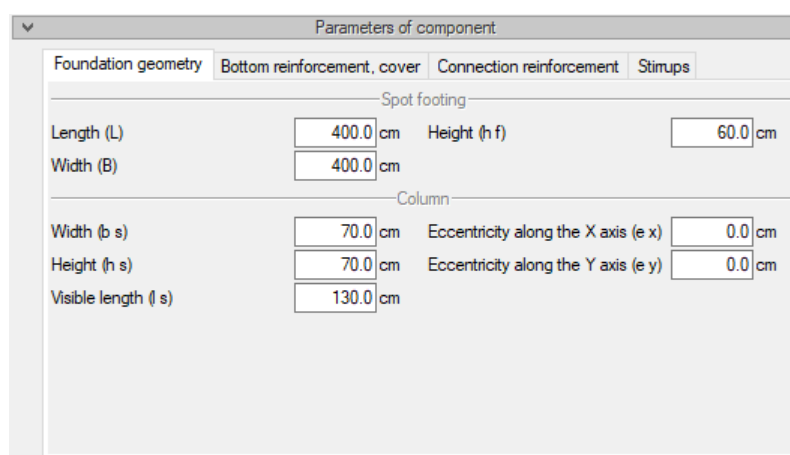


Fig. 14 A sample panel containing one row, which is divided directly into tabs with rows and columns

### 3.5. The Column

A subordinate element of each row are columns that allow for vertical separation of controls. The columns directly contain the control definitions.

#### Column tag properties

Nazwa Name		Opis Description
Id		id values
column	widthPercent	Width
params		The parameter marker

#### Example

The source code of the script with the division of the row into two columns:

```

;#xml# <row id="0">
;#xml#   <name>Spot footing</name>

```

## Defining the layout of the properties window

```

;#xml#      <columns>
;#xml#      <column id="0">
;#xml#      <widthPercent>50.0000</widthPercent>
;#xml#      <params>
; The definition of parameters
;#xml#      </params>
;#xml#      </column>
;#xml#      <column id="1">
;#xml#      <widthPercent>50.0000</widthPercent>
;#xml#      <params>
;The definition of parameters
;#xml#      </params>
;#xml#      </column>
;#xml#      </columns>
;#xml# </row>

```

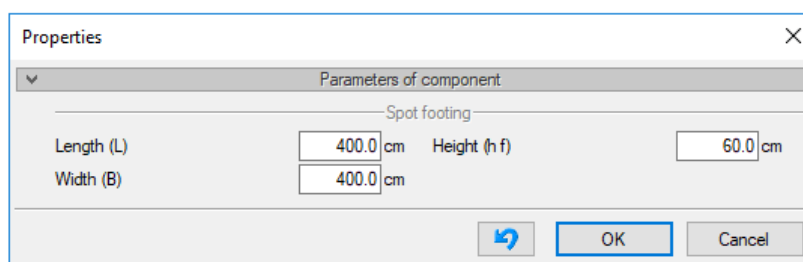


Fig. 15 A sample data panel containing one row, which is divided into two columns

### 3.6. The Pages

To divide the expanded interface into parts, you can use the so-called pages. Rows, columns and controls are defined inside the pages.

#### Page tag properties

Nazwa		Opis
id		id values
name		The text on the page bar
Image	panelId	Id of the picture panel connected to the page

## Defining the layout of the properties window

	name	Name of the image with the extension, which will be displayed in the picture panel after selecting the page
rows		Rows tab

### Example

Fragment of the source code with pages:

```

;#xml#<panel id="3">
;#xml#         <name>Parameters of component</name>
;#xml#         <defaultFoldState>0</defaultFoldState>
;#xml#         <rows>
;#xml#             <row id="0">
;#xml#                 <pages>
;#xml#                     <page id="0">
;#xml#                         <name>Foundation geometry</name>
;#xml#                         <image>
;#xml#                             <panelId>0</panelId>
;#xml#                             <name>01.png</name>
;#xml#                         </image>
;#xml#                     </rows>
; The definition of rows, columns and controls
;#xml#                     </rows>
;#xml#                 </page>
;#xml#             <page id="1">
;#xml#                 <name>Bottom reinforcement, cover</name>
;#xml#                 <image>
;#xml#                     <panelId>0</panelId>
;#xml#                     <name>02.png</name>
;#xml#                 </image>
;#xml#                 <rows>
;The definition of rows, columns and controls
;#xml#                     </rows>
;#xml#                 </page>
;#xml#             </pages>
;#xml#         </row>
;#xml#     </rows>

```

## Defining the layout of the properties window

```
; #xml#</panel>
```

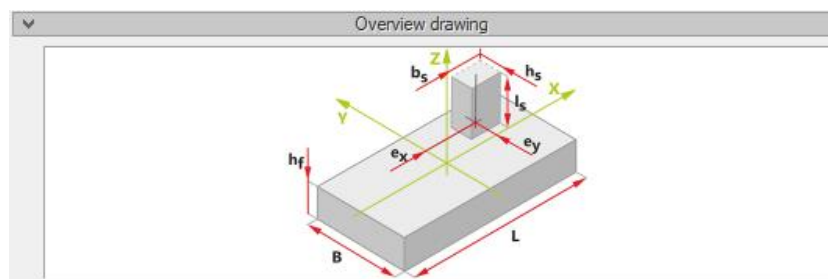


Fig. 16 A sample picture panel associated with a page

Parameters of component

Foundation geometry | Bottom reinforcement, cover | Connection reinforcement | Stirrups

Spot footing

Length (L)	400.0 cm	Height (h <sub>f</sub> )	60.0 cm
Width (B)	400.0 cm		

Column

Width (b <sub>s</sub> )	70.0 cm	Eccentricity along the X axis (e <sub>x</sub> )	0.0 cm
Height (h <sub>s</sub> )	70.0 cm	Eccentricity along the Y axis (e <sub>y</sub> )	0.0 cm
Visible length (l <sub>s</sub> )	130.0 cm		

Fig. 17 A sample panel with four pages

## 4. DEFINING CONTROLS

## Defining controls

### 4.1. Using controls

Controls are objects that are generated automatically from the loaded script code of in the ArCADia BIM system window. Examples of controls are: edit boxes, drop-down lists, check boxes, etc. Adding a control to the window takes place directly in the component's script between the tags; `;<xml#<dialog>` and `;<xml#</dialog>`. Each control has its properties related to stored values and formatting. The control is defined between the markers; `;<xml#<param>` and `</param>`. Setting the param tags directly underneath each other will cause the controls to be set automatically one after the other.

### 4.2. The Editing Field

Data input by the user takes place through editing fields, which can be appropriately defined.

**Param tag properties**

Name		Description
id		id value
name		Associating a control with the name of the variable used in the executable code
label	widthPercent	The width of the text describing the control
	name	Text describing the control
	tooltip	Text displayed in the "cloud" after moving the mouse on the text description of the control
edit	widthPercent	The width of the editing part
	type	The type of control value (int, float, string)
	value	The value transmitted (int, float, string)
	defaultValue	The default value to be restored in the control
	valueRange	The range of limit values
unit	name	Name of the unit
	widthPercent	The width of the text describing the unit



## Defining controls

### Example

Fragment of the Lisp source code for the edit type control in the ArCADia BIM system:

```
;#xml#<column id="0">
;#xml# <widthPercent>50.0000</widthPercent>
;#xml# <params>
;#xml#     <param id="0">
;#xml#         <name>fyk</name>
;#xml#         <label>
;#xml#             <widthPercent>60</widthPercent>
;#xml#             <name>Yield stress of steel (fyk)</name>
;#xml#             <tooltip>Reinforcing steel</tooltip>
;#xml#         </label>
;#xml#         <edit>
;#xml#             <widthPercent>23</widthPercent>
;#xml#             <type>float</type>
;#xml#             <value>200.0000</value>
;#xml#             <defaultValue>500.0000</defaultValue>
;#xml#             <valueRange>100.0;10000.0</valueRange>
;#xml#         </edit>
;#xml#         <unit>
;#xml#             <name>MPa</name>
;#xml#             <widthPercent>10</widthPercent>
;#xml#         </unit>
;#xml#     </param>
;the definition of subsequent parameters
;#xml# </params>
;#xml#</column>
```



Fig. 18 A sample of an Edit type control

## Defining controls

**4.3. The Drop-down Lists**

Drop-down lists are controls containing a list of value selections. Drop-down lists are divided into two types: with an editable and non-editable value selected from the list.

**Param tag for an editable list properties**

Name		Description
id		id value
name		Associating a control with the name of the variable used in the executable code
label	widthPercent	The width of the text describing the control
	name	Text describing the control
	tooltip	Text displayed in a "cloud" after moving the mouse over the text description of the control
combo	widthPercent	The width of the editing part
	type	The type of control value (int, float, string)
	readOnly	Read-only value (1 - read only, 0 - editable)
	value	The transmitted value
	defaultValue	The default value to be restored in the control
	predefinedValue	Definition of values displayed in the drop-down list
	valueRange	Range of limit values
unit	name	Unit name
	widthPercent	The width of a text describing the unit

## Defining controls

## Example

A fragment of the source code containing the definition of an editable drop-down list:

```
;#xml#<column id="0">
;#xml#      <widthPercent>50.0000</widthPercent>
;#xml#<param id="0">
;#xml#<name>fi_L</name>
;#xml#<label>
;#xml#      <widthPercent>60</widthPercent>
;#xml#      <name>Diameter (fi##L##)</name>
;#xml#      </label>
;#xml#      <combo>
;#xml#          <widthPercent>23</widthPercent>
;#xml#          <type>float</type>
;#xml#          <readOnly>0</readOnly>
;#xml#          <value>12.0000</value>
;#xml#          <defaultValue>8.0000</defaultValue>
;#xml#          <predefinedValues>5.0;5.5;6.0;8.0;10.0</predefinedValues>
;#xml#          <valueRange>(0.0000;1000.0000)</valueRange>
;#xml#      </combo>
;#xml#      <unit>
;#xml#          <name>mm</name>
;#xml#</unit>
;#xml#</param>
;definition of subsequent parameters
;#xml#      </params>
;#xml#</column>
```

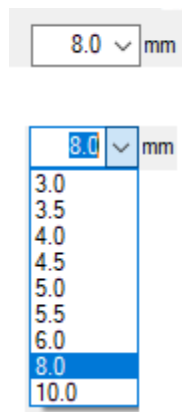


Fig. 19 The example of the editable drop-down list type control

## Defining controls

## Param tag for a non-editable list properties

Nazwa Name		Opis Description
id		id value
name		Associating a control with the name of the variable used in the executable code
label	widthPercent	The width of the text describing the control
	name	Text describing the control
	tooltip	Text displayed in a "cloud" after moving the mouse over the text description of the control
mappedCombo	widthPercent	The width of the value selection part
	type	The type of control value (int, float)
	value	The mapped value - transmitted
	defaultValue	The default value to be restored in the control
	valueRange	The map of values, i.e. values and keys
unit	name	Unit name
	widthPercent	The width of a text describing the unit

## Example

A fragment of the source code containing the definition of a non-editable drop-down list:

```

;#xml#<param id="0">
;#xml# <name>concrete_class</name>
;#xml# <label>
;#xml#     <widthPercent>30</widthPercent>
;#xml#     <name>Class of concrete</name>
;#xml# </label>
;#xml# <mappedCombo>
;#xml#     <widthPercent>28</widthPercent>
;#xml#     <type>int</type>

```

## Defining controls

```

;#xml#      <value>60</value>
;#xml#      <defaultValue>12</defaultValue>
;#xml#      <valueRange>12;C12/15;16;C16/20;20 </valueRange>
;#xml# </mappedCombo>
;#xml# <unit>
;#xml#      <name></name>
;#xml# </unit>
;#xml#</param>

```

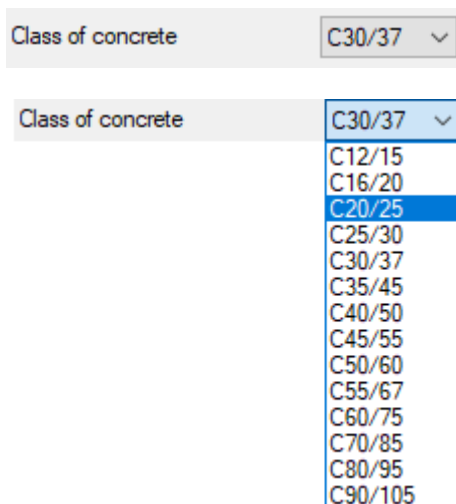


Fig. 20 The example of the non-editable drop-down list type control

## 4.4. The radioGroup type button

The grouped buttons are created using the radioGroup control. This is a managing control that allows the user to select only one of the predefined set of mutually exclusive options.

### Param tag properties

Nazwa Name		Opis Description
id		id value
name		Associating a control with the name of the variable used in the executable code
label	widthPercent	The width of the text describing the control
	Name	Option texts to be separated by ";"
radioGroup	Value	The transmitted value corresponding to the number of selected option

## Defining controls

	defaultValue	The default value to be restored in the control
--	--------------	---

### Example

Fragment of the Lisp source code for the ArCADia BIM system for the radioGroup control:

```
;#xml#<column id="0">
;#xml# <widthPercent>50.0000</widthPercent>
;#xml# <params>
;#xml#      <param id="1">
;#xml#          <name>stirrup</name>
;#xml#          <label>
;#xml#              <widthPercent>30</widthPercent>
;#xml# <name>Two-cut reinforcement;Four-cut reinforcement</name>
;#xml#          </label>
;#xml#          <radioGroup>
;#xml#              <value>1</value>
;#xml#              <defaultValue>1</defaultValue>
;#xml#          </radioGroup>
;#xml#      </param>
;definition of subsequent parameters
;#xml# </params>
;#xml#</column>
```

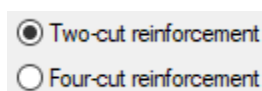


Fig. 21 An example of the radioGroup control

## 4.5. The checkbox type button

The checkbox field is a control that allows you to make a binary selection. The choice of value takes place between two mutually exclusive possibilities.

### Param tag properties

Nazwa Name	Opis Description
id	id value

## Defining controls

	name	Associating a control with the name of the variable used in the executable code
label	widthPercent	The width of the text describing the control
	name	Text describing the control
	tooltip	Text displayed in a "cloud" after moving the mouse over the text description of the control
checkbox	value	The transmitted value
	defaultValue	The default value to be restored in the control

## Example

Fragment of the Lisp source code for the ArCADia BIM system for the edit type control:

```
;#xml#<column id="0">
;#xml# <widthPercent>50.0000</widthPercent>
;#xml# <params>
;#xml#     <param id="1">
;#xml#         <name>reb_add</name>
;#xml#         <label>
;#xml#             <widthPercent>30</widthPercent>
;#xml#             <name>Insert additional reinforcement</name>
;#xml#         </label>
;#xml#         <checkbox>
;#xml#             <value>1</value>
;#xml#             <defaultValue>1</defaultValue>
;#xml#         </checkbox>
;#xml#     </param>
; definition of subsequent parameters
;#xml# </params>
;#xml#</column>
```

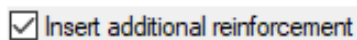


Fig. 22 A sample of a checkbox type control

## 5. AVAILABLE FEATURES OF THE ARCADIA BIM SYSTEM



## 5.1. Dialogue and decision window

- Command:  
(**command "rcc\_sm" komunikat typ**)
- Description:  
The result of calling the function is displaying a dialog with a specific message.
- Parameters:
  - **message** - the content of the displayed message,
  - **Type** – the window type. Accepted values: 0 - information dialog box, 1 - warning dialog box, 2 - error dialog box.

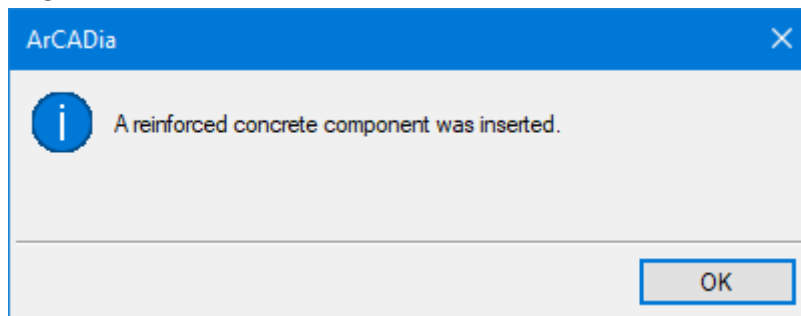


Fig. 23 A dialog box - information

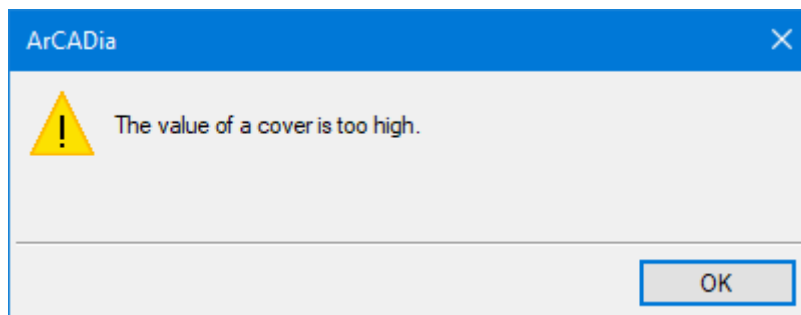


Fig. 24 A dialog box - warning

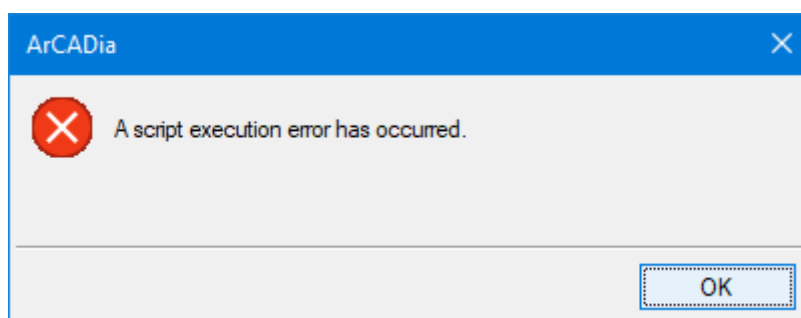


Fig. 25 A dialog box - error

## Available features of the ArCADia BIM system

- A script sample:

An example of the source code using the given commands:

```
;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#

(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)

(command "rcc_sm" "A reinforced concrete component was
inserted." 0)
(command "rcc_sm" "The value of a cover is too high." 1)
(command "rcc_sm" "A script execution error has occurred." 2)
```

## 5.2. Inserting components

### 5.2.1. The empty 3D object

- Command:

(**command** "rcc\_object3d\_io" name k h)

- Description:

The result of calling the function is registering a new 3D object in the Project Manager with given name, hatch number and concrete class. Calling this function is necessary for each script.

- Parameters

**name** - the name of the object displayed after insertion in the Project Manager tree,  
**k**- the concrete class. Adopted values: 12, 16, 20, 25, 30, 35, 40, 45, 50, 55, 60, 70, 80, 90.  
The adopted value depends on the concrete class, e.g. C12/15 → k = 12, C16/20 → k = 16, C20/25 → k = 20, C25/30 → k = 25, C30/37 → k = 30, C35/45 → k = 35, C40/50 → k = 40, C45/55 → k = 45, C50/60 → k = 50, C55/67 → k = 55, C60/75 → k = 60, C70/85 → k = 70, C80/95 → k = 80, C90/105 → k = 90,  
**h**- the hatch no. Adopted values: empty → h = 2000, ANSI\_3 → h = 2008, ANSI → h = 2004\_33, stipple → h = 2002.

- A function example:

(**command** "rcc\_object3d\_io" "Strip footing" 20 20000)

- A script example:

A sample of the source code inserting an empty 3D object into the project:

## Available features of the ArCADia BIM system

```

;#xml#<dialog>
;#xml# <panels>
;#xml# </panels>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_sm" "Algorithm made correctly." 0)

```

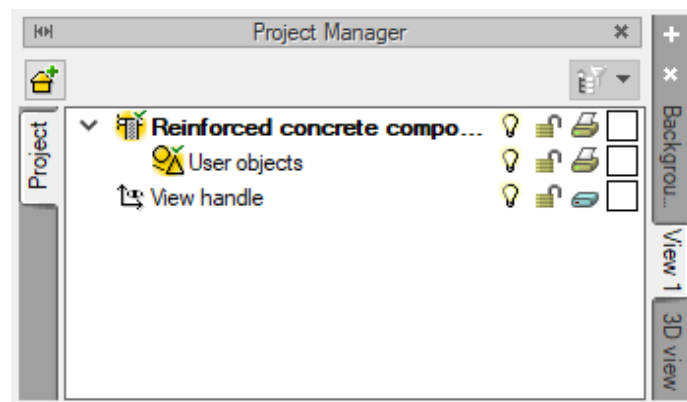


Fig. 26 A new 3D object created after the script is properly run - Project Manager

### 5.2.2. Adding a cuboid

- Command:

(**command** "rcc\_object3d\_aco" (*list* *x y z*) *L B H*)

- Description:

The result of calling the function is inserting a cuboid in a specified location and specified sizes.

- Parameters:

*x y z* –the coordinates of the cuboid's insertion point,

*L* –the dimension of the cuboid on the x axis,

*B* –the dimension of the cuboid on the y axis,

*H* –the dimension of the cuboid on the z axis.

- A function example:

(**command** "rcc\_object3d\_aco" (*list* 0 0 0) 200 300 50)

- A script sample:

An example of the source code - creation of a cuboid solid:

```

;#xml#<dialog>

```

## Available features of the ArCADia BIM system

```

;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 300 200 50)
(command "rcc_sm" "Algorithm made correctly. " 0)

```

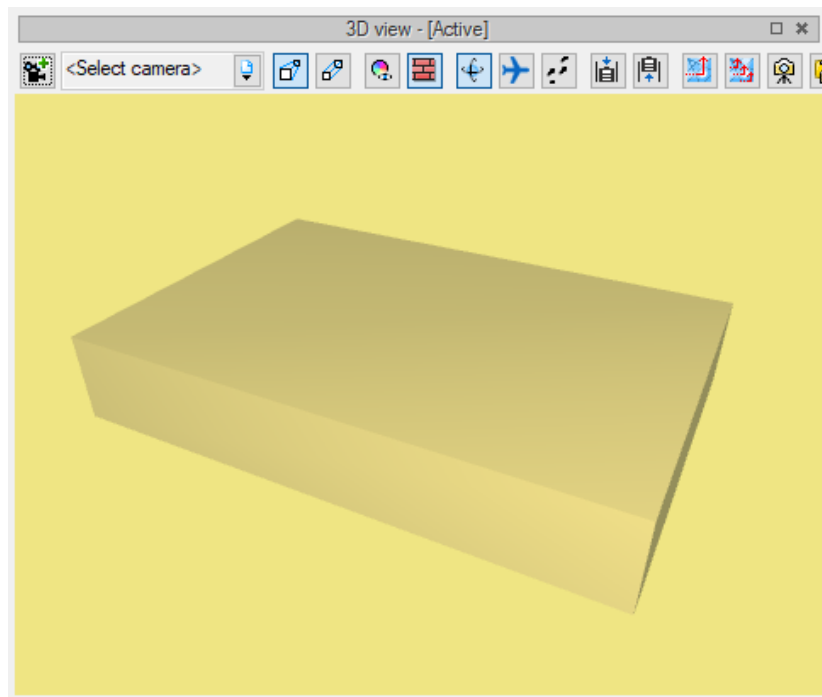


Fig. 27 3D view of the created solid

We can combine any solids in the scripts, which results in creating single merged solid.

An example of the source code inserting two solids that will merge into a single solid:

```

;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 300 200 50)
(command "rcc_object3d_isv_north" (list 100 0 0))

```

## Available features of the ArCADia BIM system

```
(command "rcc_sm" "Algorithm made correctly." 0)
```

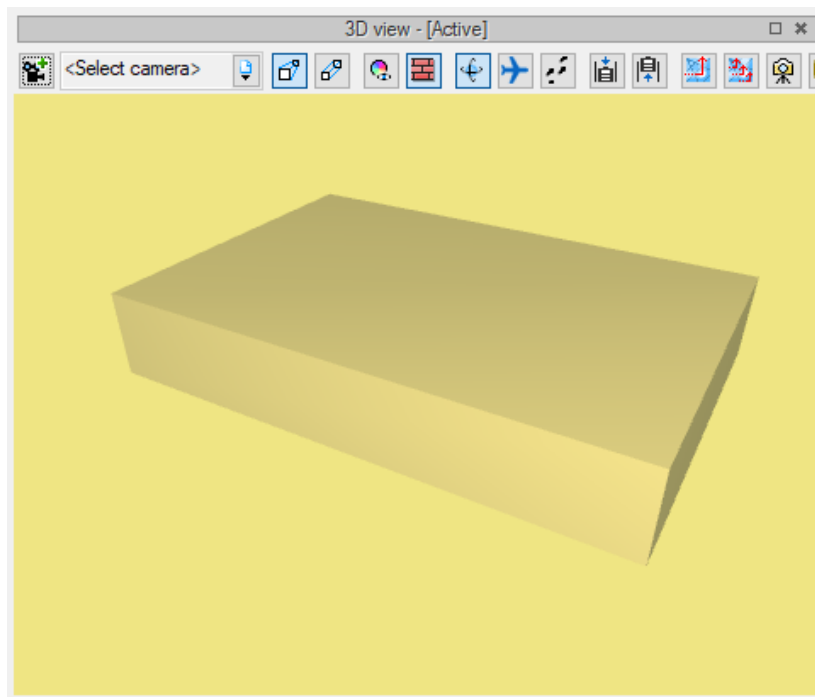


Fig. 28 3D view of the merged solid

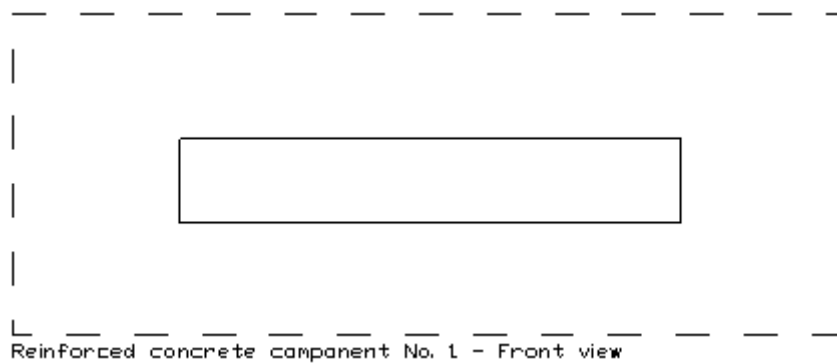


Fig. 29 A front view of the merged solids in the work area

### 5.2.3. Inserting a cuboid

- Command:

```
(command "rcc_object3d_ieo" name h (list x y z) L B H)
```

- Description:

The result of calling the function is inserting a cuboid in the given location, with specified dimensions, hatch number and concrete class, which will not be merged with other existing solids in the project.

**Available features of the ArCADia BIM system**

- Parameters:

**name** - name of the object,

**h** - hatch number. Adopted values: empty → h = 2000, ANSI\_3 → h = 2008, ANSI → h = 2004 \_33, stipple → h = 2002,

**x y z** –the coordinates of the cuboid's insertion point,

**L** –the dimension of the cuboid on the x axis,

**B** –the dimension of the cuboid on the y axis,

**H** –the dimension of the cuboid on the z axis.

- A function example:

(**command** "rcc\_object3d\_ieo" "object nr 2" 20000 (list 0 0 50) 300 200 50)

- A script example:

An example of the source code that inserts two cuboids solids that will not merge:

```
;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 300 200 50)
(command "rcc_object3d_ieo" "object nr 2" 20008 (list 0 0 50) 300 200
50)
(command "rcc_object3d_isv_north" (list 100 0 0))
(command "rcc_sm" "Algorithm made correctly." 0)
```

## Available features of the ArCADia BIM system

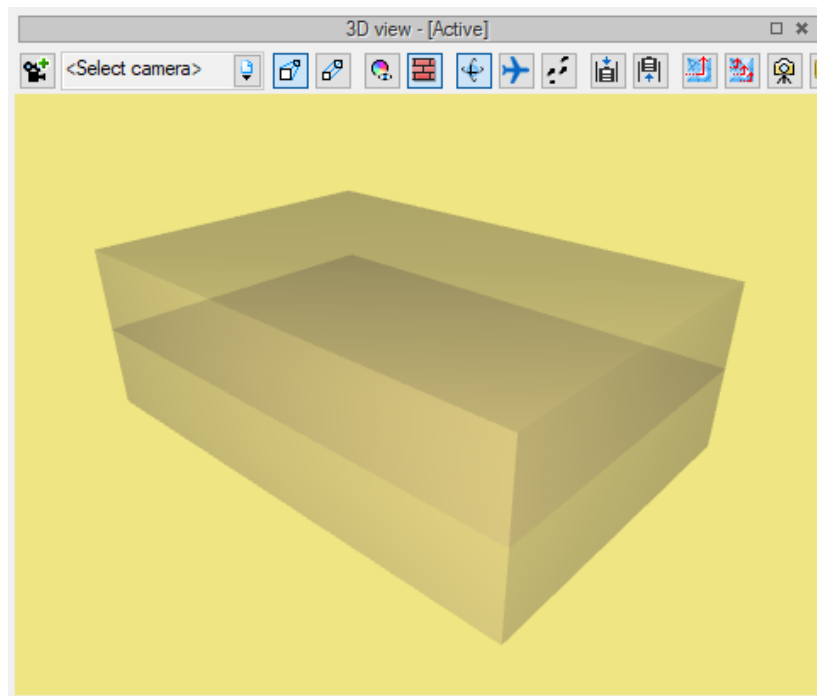


Fig. 30 3D view – failure of two solids merging after execution of the script

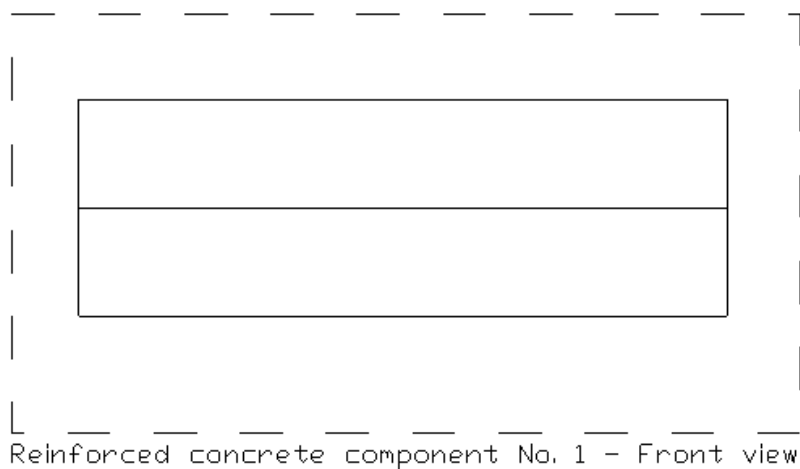


Fig. 31 A front view in the work area - no merging of two solids

#### 5.2.4. Adding a cylinder

- Command:  
(**command** "rcc\_object3d\_aeco" (*list*  $x_1$   $y_1$   $z_1$ ) (*list*  $x_2$   $y_2$   $z_2$ )  $r$ )
- Description:  
The result of calling the function is the insertion of a cylinder with a given diameter and the location of the center points of the bases.
- Parameters:

## Available features of the ArCADia BIM system

$x_1 y_1 z_1$  –the coordinates of the center point of the first cylinder base,  
 $x_2 y_3 z_4$  –the coordinates of the center point of the second cylinder base,  
 $r$  –the radius of the cylinder base.

- A function example:

(command "rcc\_object3d\_aeco" (list 0 0 0) (list 0 0 200) 100)

- A script example:

An example of the source code - creation of a cylinder body:

```
;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aeco" (list 0 0 0) (list 0 0 200) 100)
(command "rcc_sm" "Algorithm made correctly." 0)
```

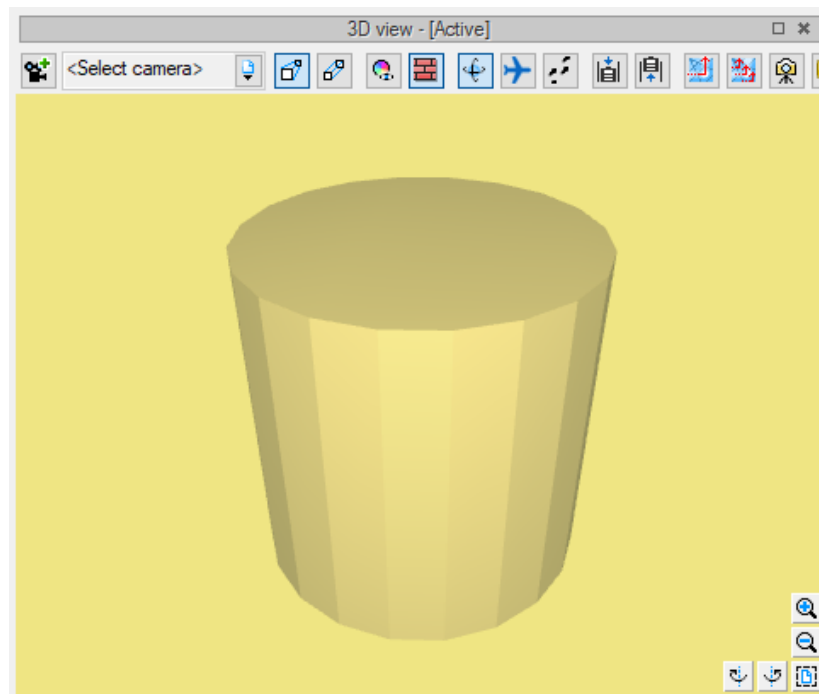


Fig. 32 3D view of the created solid



### 5.2.5. Adding an extruded solid

- Command:

(**command** "rcc\_object3d\_aeo" (*list*  $x_1$   $y_1$   $z_1$ ) (*list*  $x_2$   $y_2$   $z_2$ )  $n_w$   $p_1 p_2 \dots p_{n_w}$

$n_d$   $n_{wd}$   $p_1 p_2 \dots p_{n_d}$ )

- Description:

The result of calling the function is inserting an extruded solid with a given position and any base geometry. An openings of any shape can be cut in the solid.

- Parameters:

$x_1 y_1 z_1$  –the coordinates of the point containing the plane of the first base of the solid,

$x_2 y_2 z_2$  –the coordinates of the point containing the plane of the second base of the solid,

$n_w$  –the number of vertices of the solids base,

$p_1 p_2 \dots p_{n_w}$  –the coordinates of the contour points of the solid base. A single point is saved using a list containing three numbers defining the coordinates (x y z), e.g. (list 0 0 0),

$n_d$  –the number of openings in the solid,

$n_{wd}$  –the number of vertices of the next opening,

$p_1 p_2 \dots p_{n_d}$  –the coordinates of the contour points of the next opening. A single point is saved using a list containing three numbers defining the coordinates (x y z), e.g. (list 0 0 0),

- A function example:

(**command** "rcc\_object3d\_aeo" (*list* 0 0 0) (*list* 0 0 100) 3 (*list* 0 0 0)

(*list* 100 0 0) (*list* 50 50 0) 0)

- A script example:

An example of the source code - creation of an extruded solid:

```
;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aeo" (list 0 0 0) (list 0 0 100) 3 (list
0 0 0)
(list 100 0 0) (list 50 50 0) 0)
(command "rcc_sm" "Algorithm made correctly." 0)
```

## Available features of the ArCADia BIM system

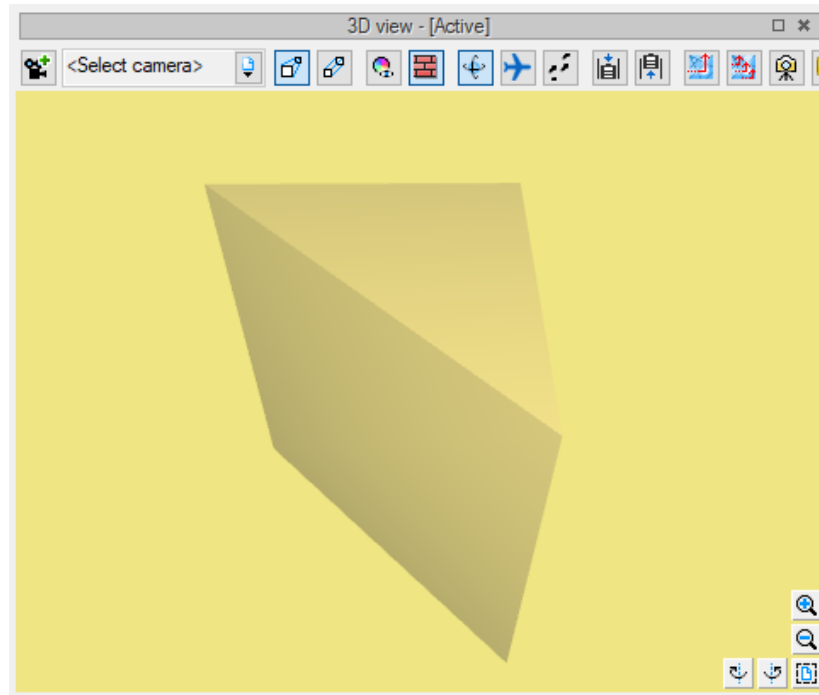


Fig. 33 3D view of the created solid

## 5.2.6. Adding the cutting of the solid to the concrete

- Command:

(**command** "rcc\_object3d\_aocf"  $n_w$   $p_1 p_2 \dots p_{n_w}$ )

- Description:

The result of calling the function is inserting a horizontal line of the solid cutting on views and cross-sections.

- Parametrs:

$n_w$  –the number of vertices of the contour plane, which will contain the "cut off" of the extreme cross-section of the solid,

$p_1 p_2 \dots p_{nd}$  –the coordinates of the cut off contour points. A single point is saved using a list containing three numbers defining the coordinates (x y z), e.g. (list 0 0 0),

- A function example:

(**command** "rcc\_object3d\_aocf"

4 (list 0 0 200) (list 100 0 200) (list 100 200 200) (list 0 200 200))

- A script example:

An example of the source code - insertion of the cutting:

```
;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
```

**Available features of the ArCADia BIM system**

```
;#lispParams#  
(command "rcc_object3d_io" "Reinforced concrete component No. 1"  
30 2000)  
(command "rcc_object3d_aco" (list 0 0 0) 50 50 200)  
(command "rcc_object3d_aocf" 4 (list 0 0 200) (list 100 0 200)  
(list 100 200 200) (list 0 200 200))  
(command "rcc_object3d_isv_north" (list 100 0 0))  
(command "rcc_sm" "Algorithm made correctly." 0)
```

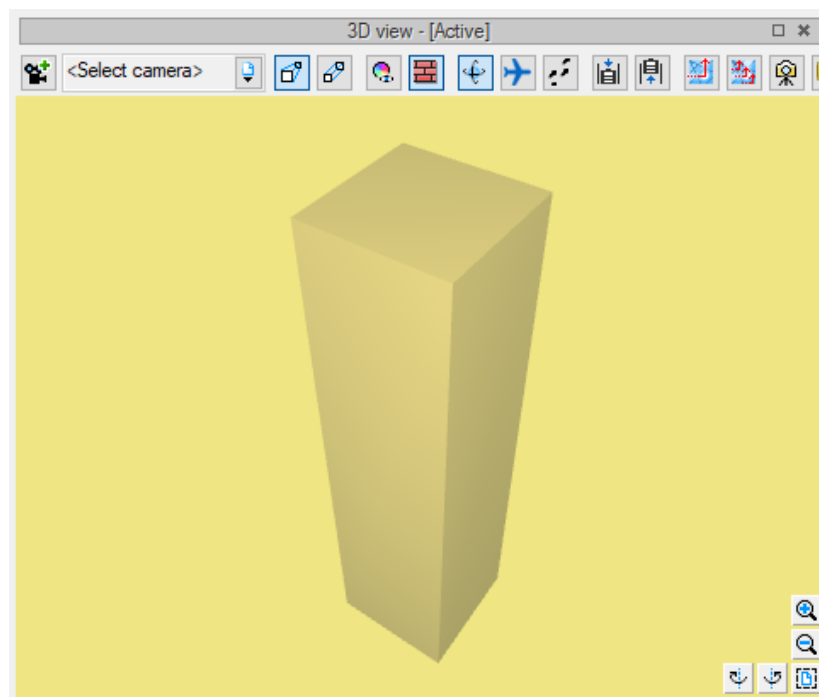


Fig. 34 3D view of the created solid

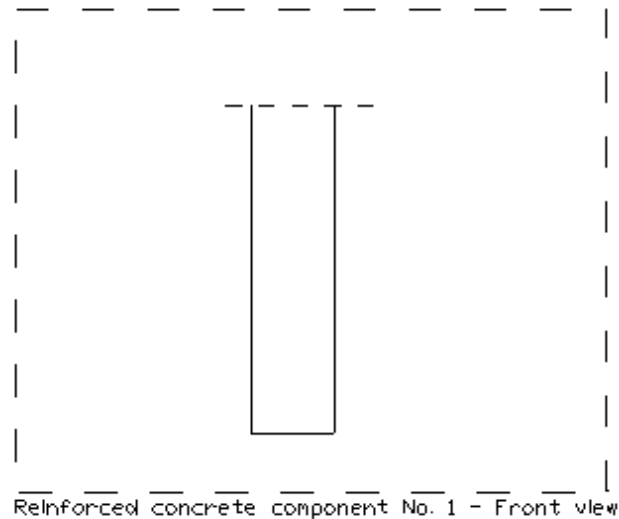


Fig. 35 The front view with a visible cut line in the working area

### 5.2.7. Adding an extruded pyramid to the concrete solid

- Command:

(**command** "rcc\_object3d\_aep" (*list*  $x_1$   $y_1$   $z_1$ ) (*list*  $x_2$   $y_2$   $z_2$ )  
 $n_w$   $p_{d1}p_{d2} \dots p_{dn_w}$   $p_{g1}p_{g2} \dots p_{gn_w}$ )

- Description:

The result of calling the function is inserting a pyramid in the given location and the given dimension of the cross-sections.

- Parameters

$x_1y_1z_1$  – the coordinates of the point containing the plane of the first base of the solid,

$x_2y_2z_2$  – the coordinates of the point containing the plane of the second base of the solid,

$n_w$  – the number of vertices of the pyramid base,

$p_{d1}p_{d2} \dots p_{dn_w}$  – the coordinates of the contour points of the pyramid base. A single point is saved using a list containing three numbers defining the coordinates (x y z), e.g. (list 0 0 0),

$p_{g1}p_{g2} \dots p_{gn_w}$  – the coordinates of the cut part points. A single point is saved using a list containing three numbers defining the coordinates (x y z), e.g. (list 0 0 0),

- A function example:

```
(command "rcc_object3d_aep" (list 0 0 0) (list 0 0 100)
4 (list 0 0 0)(list 100 0 0)(list 100 100 0)(list 0 100 0)
(list 25 25 0) (list 75 25 0) (list 75 75 0) (list 25 75 0))
```

- A script example:

An example of the source code - inserting a truncated pyramid

```
; #xml# <dialog>
; #xml# <panels/>
; #xml# </dialog>
```

## Available features of the ArCADia BIM system

```

;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aep" (list 0 0 0) (list 0 0 100) 4
(list 0 0 0) (list 100 0 0) (list 100 100 0) (list 0 100 0)
(list 25 25 0) (list 75 25 0) (list 75 75 0) (list 25 75 0))
(command "rcc_sm" "Algorithm made correctly." 0)

```

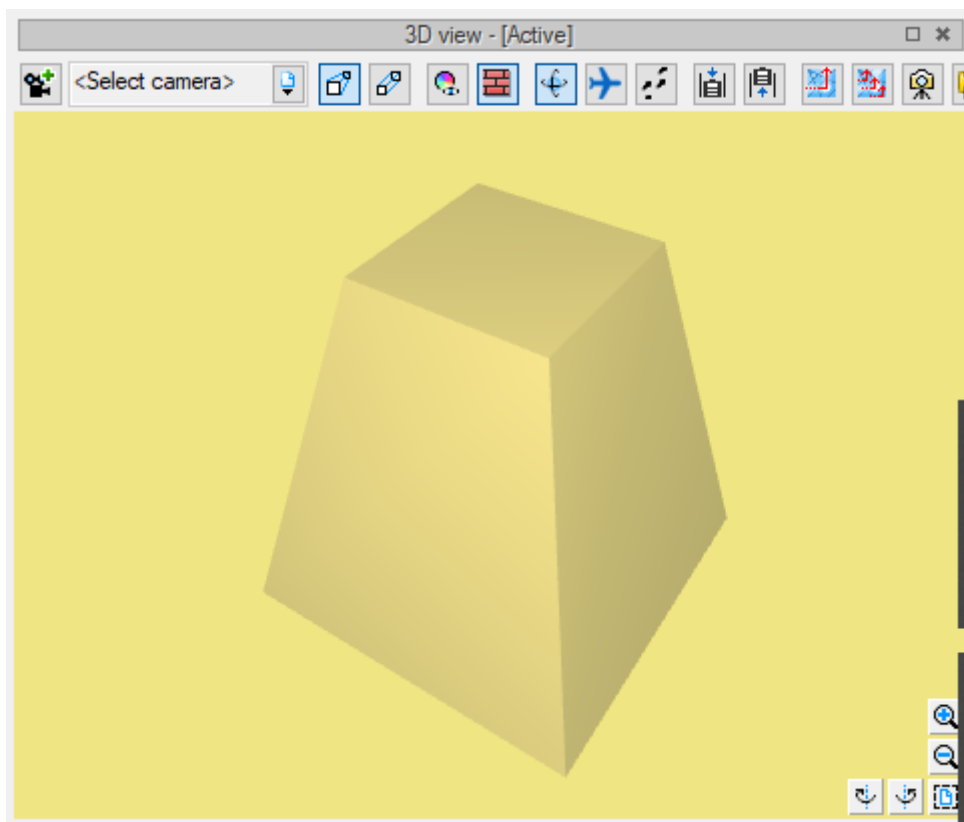


Fig. 36 3D view of the created solid

## 5.2.8. Adding an inverted extruded pyramid to a concrete solid

- Command:

(**command** "rcc\_object3d\_aerp" (*list*  $x_1$   $y_1$   $z_1$ ) (*list*  $x_2$   $y_2$   $z_2$ )  
 $n_w$   $p_{d1} p_{d2} \dots p_{dn_w}$   $h_w$   $p_{g1} p_{g2} \dots p_{gn_w}$ )

- Description:

The result of calling the function is inserting a pyramid in the given location and the given dimension of the cross-sections.

- Parameters:

**Available features of the ArCADia BIM system**

$x_1 y_1 z_1$  – the coordinates of the point containing the plane of the first base of the solid,  
 $x_2 y_2 z_2$  – the coordinates of the point containing the plane of the second base of the solid,  
 $n_w$  – the number of vertices of the contour base,  
 $p_{d1} p_{d2} \dots p_{dnw}$  – the coordinates of the base contour points. A single point is saved using a list containing three numbers defining the coordinates (x y z), e.g. (list 0 0 0),  
 $h_w$  – the depth of the pyramid,  
 $p_{g1} p_{g2} \dots p_{gnw}$  – the coordinates of the top part of the pyramid contour points. A single point is saved using a list containing three numbers defining the coordinates (x y z), e.g. (list 0 0 0),

- A function example:

```
(command "rcc_object3d_aerp" (list 0 0 60) (list 0 0 0)
4 (list 0 0 0) (list 100 0 0) (list 100 100 0) (list 0 100 0)
40 (list 25 25 20) (list 75 25 0) (list 75 75 0) (list 25 75 0))
```

- A script example:

An example of the source code - adding an inverted extruded pyramid to a concrete solid:

```
; #xml# <dialog>
; #xml# <panels/>
; #xml# </dialog>
; #lispParams#
; #lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aerp" (list 0 0 60) (list 0 0 0) 4
(list 0 0 0) (list 100 0 0) (list 100 100 0) (list 0 100 0)
40 (list 25 25 20) (list 75 25 0) (list 75 75 0) (list 25 75 0))
(command "rcc_sm" "Algorithm made correctly." 0)
```

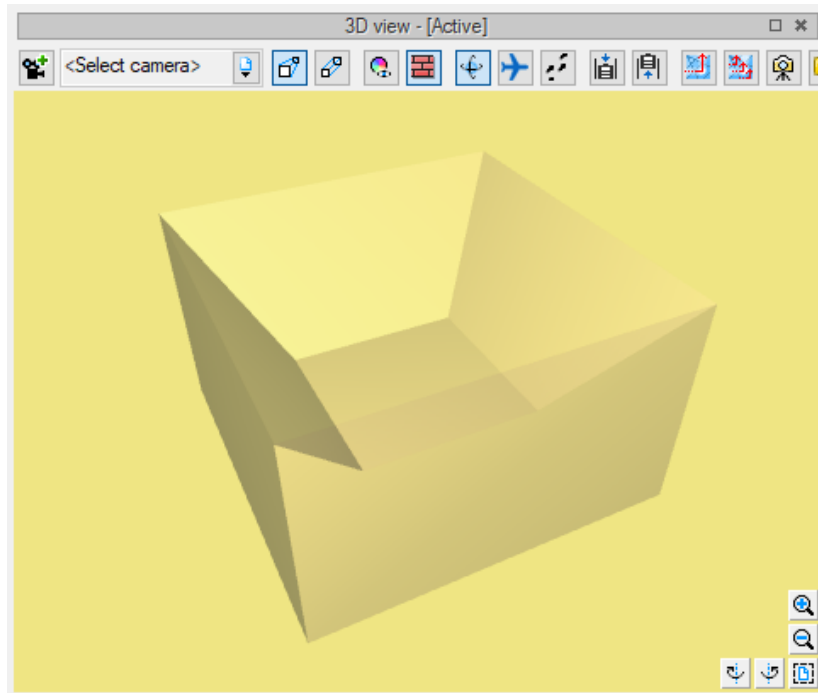


Fig. 37 3D view of the created solid

## 5.3. Inserting views

### 5.3.1. The front view

- Command:  
(**command** "rcc\_object3d\_isv\_north" (*list* **x y z**))

- Description:

The result of calling the function is inserting the view from the front of the object at the point with the given location.

- Parameters:  
**x y z** – the coordinates of the view insertion point,
- A function example:  
(**command** "rcc\_object3d\_isv\_north" (*list* 0 0 0))

- A script example:

```
;#xml#<dialog>
;#xml#  <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
```

## Available features of the ArCADia BIM system

```
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)

(command "rcc_object3d_aco" (list 0 0 0) 300 200 50)
(command "rcc_object3d_isv_north" (list 0 0 0))
(command "rcc_sm" "Algorithm made correctly." 0)
```

### 5.3.2. The rear view

- Command:  
(**command** "rcc\_object3d\_isv\_south" (*list* x y z))
- Description:  
The result of calling the function is inserting the view from the back of the object at the point with the given location.
- Parameters:  
x y z – the coordinates of the view insertion point,
- A function example :  
(**command** "rcc\_object3d\_isv\_south" (*list* 0 0 0))
- A script example:

```
;#xml#<dialog>
;#xml#  <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1" 30
2000)
(command "rcc_object3d_aco" (list 0 0 0) 300 200 50)
(command "rcc_object3d_isv_south" (list 0 0 0))
(command "rcc_sm" "Algorithm made correctly." 0)
```

### 5.3.3. The view from the left

- Command:  
(**command** "rcc\_object3d\_isv\_east" (*list* x y z))
- Description:



### Available features of the ArCADia BIM system

The result of calling the function is inserting the view from the left side of the object at the point with the given location.

- Parameters:

**x y z** –the coordinates of the view insertion point,

- A function example:

(**command** "rcc\_object3d\_isv\_east" (**list** 0 0 0))

- A script example:

```
;#xml#<dialog>
;#xml#  <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 300 200 50)
(command "rcc_object3d_isv_east" (list 0 0 0))
(command "rcc_sm" "Algorithm made correctly." 0)
```

#### 5.3.4. The view from the right

- Command:

(**command** "rcc\_object3d\_isv\_west" (**list** x y z))

- Description:

The result of calling the function is inserting the view from the right side of the object at the point with the given location.

- Parameters:

**x y z** – the coordinates of the view insertion point,

- A function example:

(**command** "rcc\_object3d\_isv\_west" (**list** 0 0 0))

- A script example:

```
;#xml#<dialog>
;#xml#  <panels/>
;#xml#</dialog>
;#lispParams#
```

## Available features of the ArCADia BIM system

```
;#lispParams#  
(command "rcc_object3d_io" "Reinforced concrete component No. 1"  
30 2000)  
(command "rcc_object3d_aco" (list 0 0 0) 300 200 50)  
(command "rcc_object3d_isv_west" (list 0 0 0))  
(command "rcc_sm" "Algorithm made correctly." 0)
```

## 5.4. Inserting cross-sections

### 5.4.1. The horizontal cross-section directed downwards

- Command:

(**command "rcc\_object3d\_icsv\_ground"** (*list*  $x_1$   $y_1$   $z_1$ ) (*list*  $x_2$   $y_2$   $z_2$ ) *dv*)

- Description:

The result of calling the function is inserting a horizontal cross-section directed downwards at the point with a given location in the object and inserting it at the specific point in the work area.

- Parameters:

$x_1$   $y_1$   $z_1$  – the coordinates of the point of cross-section execution,

$x_2$   $y_2$   $z_2$  – the coordinates of the insertion point of the cross-section view,

*dv* – the depth of reinforcing bars visibility and the cross-section contours.

- A function example:

(**command "rcc\_object3d\_icsv\_ground"** (*list* 100 50 50) (*list* 0 0 0) 40.0)

- A script example:

```
;#xml#<dialog>  
;#xml#  <panels/>  
;#xml#</dialog>  
;#lispParams#  
;#lispParams#  
(command "rcc_object3d_io" "Reinforced concrete component No. 1"  
30 2000)  
(command "rcc_object3d_aco" (list 0 0 0) 200 100 100)  
(command "rcc_object3d_icsv_ground" (list 100 50 50) (list 0 0 0)  
40.0)  
(command "rcc_sm" "Algorithm made correctly." 0)
```

### 5.4.2. Horizontal cross-section directed upwards

- Command:

(**command** "rcc\_object3d\_icsv\_sky" (*list*  $x_1 y_1 z_1$ ) (*list*  $x_2 y_2 z_2$ )  $dv$ )

- Description:

The result of calling the function is inserting a horizontal cross-section directed upwards at the point with a given location in the object and inserting it at the specific point in the work area.

- Parameters:

$x_1 y_1 z_1$  – the coordinates of the point of cross-section execution,

$x_2 y_2 z_2$  – the coordinates of the insertion point of the cross-section view,

$dv$  – the depth of reinforcing bars visibility and the cross-section contours.

- A function example:

(**command** "rcc\_object3d\_icsv\_sky" (*list* 100 50 50) (*list* 0 0 0) 40.0)

- A script example:

```
;#xml#<dialog>
;#xml#  <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 200 100 100)
(command "rcc_object3d_icsv_sky" (list 100 50 50) (list 0 0 0)
40.0)
(command "rcc_sm" "Algorithm made correctly." 0)
```

### 5.4.3. The vertical cross-section directed to the right

- Command:

(**command** "rcc\_object3d\_icsv\_east" (*list*  $x_1 y_1 z_1$ ) (*list*  $x_2 y_2 z_2$ )  $dv$ )

- Description:

The result of calling the function is inserting a vertical cross-section directed to the right at the point with a given location in the object and inserting it at the specific point in the work area.

- Parameters:

$x_1 y_1 z_1$  – the coordinates of the point of cross-section execution,

$x_2 y_2 z_2$  – the coordinates of the insertion point of the cross-section view,

## Available features of the ArCADia BIM system

***dv*** – the depth of reinforcing bars visibility and the cross-section contours.

- function example:

(**command** "rcc\_object3d\_icsv\_east" (*list* 100 50 50) (*list* 0 0 0) 40.0)

- A script example:

```
;#xml#<dialog>
;#xml#  <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 200 100 100)
(command "rcc_object3d_icsv_east" (list 100 50 50) (list 0 0 0)
40.0)
(command "rcc_sm" "Algorithm made correctly." 0)
```

#### 5.4.4. The vertical cross-section directed to the left

- Command:

(**command** "rcc\_object3d\_icsv\_west" (*list*  $x_1$   $y_1$   $z_1$ ) (*list*  $x_2$   $y_2$   $z_2$ ) *dv*)

- Description:

The result of calling the function is inserting a vertical cross-section directed to the left at the point with a given location in the object and inserting it at the specific point in the work area.

- Parameters:

$x_1$   $y_1$   $z_1$  – the coordinates of the point of cross-section execution,

$x_2$   $y_2$   $z_2$  – the coordinates of the insertion point of the cross-section view,

***dv*** – the depth of reinforcing bars visibility and the cross-section contours.

- A function example:

(**command** "rcc\_object3d\_icsv\_west" (*list* 100 50 50) (*list* 0 0 0) 40.0)

- A script example:

```
;#xml#<dialog>
;#xml#  <panels/>
;#xml#</dialog>
;#lispParams#
```

## Available features of the ArCADia BIM system

```
;#lispParams#  
(command "rcc_object3d_io" "Reinforced concrete component No. 1"  
30 2000)  
(command "rcc_object3d_aco" (list 0 0 0) 200 100 100)  
(command "rcc_object3d_icsv_west" (list 100 50 50) (list 0 0 0)  
40.0)  
(command "rcc_sm" "Algorithm made correctly." 0)
```

### 5.4.5. The vertical cross-section directed to the front

- Command:

(**command** "rcc\_object3d\_icsv\_north" (*list*  $x_1$   $y_1$   $z_1$ ) (*list*  $x_2$   $y_2$   $z_2$ ) *dv*)

- Description:

The result of calling the function is inserting a vertical cross-section directed to the front at a point with a given location in the object and inserting it at a specific point in the work area.

- Parameters:

$x_1$   $y_1$   $z_1$  – the coordinates of the point of cross-section execution,

$x_2$   $y_2$   $z_2$  – the coordinates of the insertion point of the cross-section view,

*dv* – the depth of reinforcing bars visibility and the cross-section contours.

- A function example:

(**command** "rcc\_object3d\_icsv\_north" (*list* 100 50 50) (*list* 0 0 0) 40.0)

- A script example:

```
;#xml#<dialog>  
;#xml# <panels/>  
;#xml#</dialog>  
;#lispParams#  
;#lispParams#  
(command "rcc_object3d_io" "Reinforced concrete component No. 1"  
30 2000)  
(command "rcc_object3d_aco" (list 0 0 0) 200 100 100)  
(command "rcc_object3d_icsv_north" (list 100 50 50) (list 0 0 0)  
40.0)  
(command "rcc_sm" "Algorithm made correctly." 0)
```

### 5.4.6. The vertical cross-section directed to the back

- Command:

(**command** "rcc\_object3d\_icsv\_south" (*list*  $x_1$   $y_1$   $z_1$ ) (*list*  $x_2$   $y_2$   $z_2$ ) *dv*)

- Description:

The result of calling the function is inserting a vertical cross-section directed to the back at a point with a given location in the object and inserting it at a specific point in the work area.

- Parameters:

$x_1$   $y_1$   $z_1$  – the coordinates of the point of cross-section execution,

$x_2$   $y_2$   $z_2$  – the coordinates of the insertion point of the cross-section view,

*dv* – the depth of reinforcing bars visibility and the cross-section contours.

- A function example:

(**command** "rcc\_object3d\_icsv\_south" (*list* 100 50 50) (*list* 0 0 0) 40.0)

- A script example:

```
;#xml#<dialog>
;#xml#  <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 200 100 100)
(command "rcc_object3d_icsv_south" (list 100 50 50) (list 0 0 0)
40.0)
(command "rcc_sm" "Algorithm made correctly." 0)
```

### 5.4.7. Sample results of the script inserting selected cross-sections

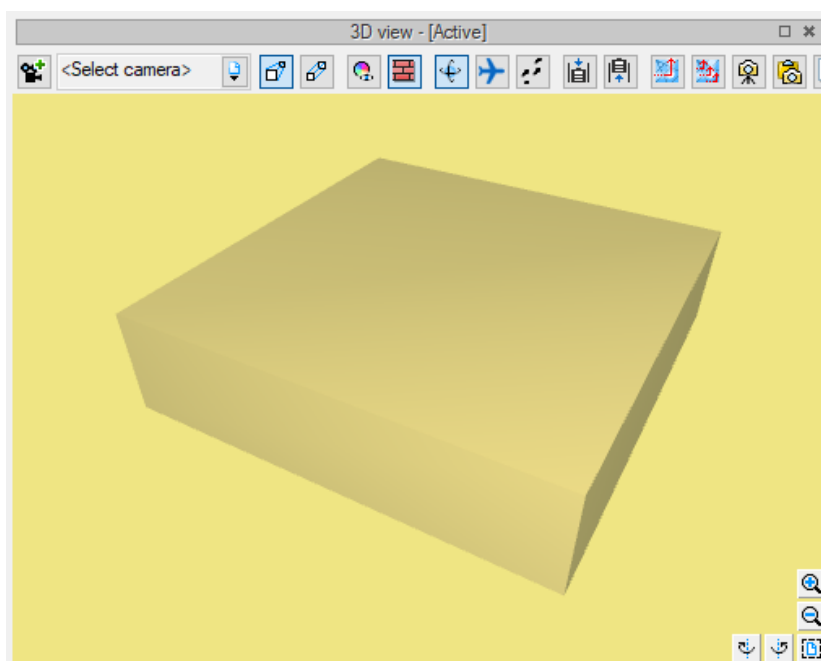


Fig. 38 The 3D solid view

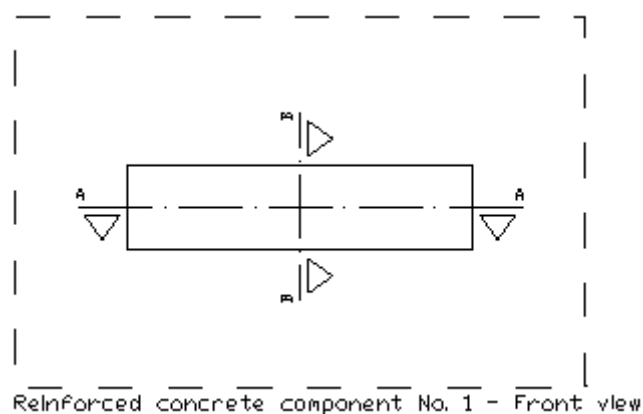


Fig. 39 The front view of the solid in the project work area with lines symbolizing cross-sections

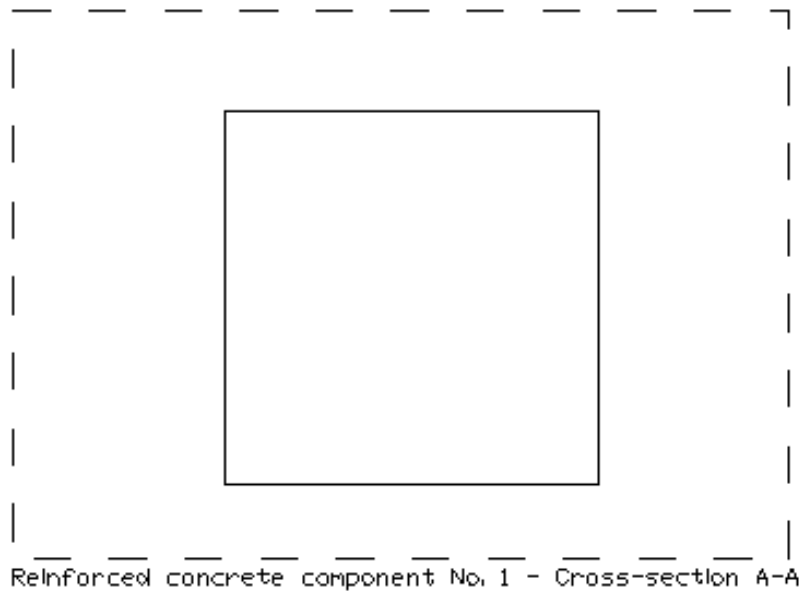


Fig. 40 The A-A cross-section in the working area

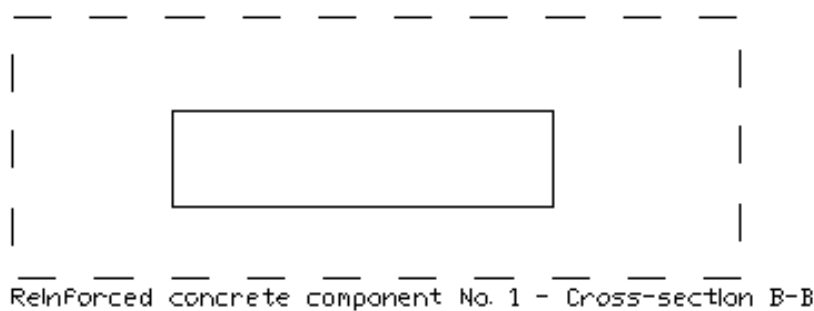


Fig. 41 The B-B cross-section in the working area

## 5.5. Inserting reinforcement

### 5.5.1. The rebars

- Command:

(**command** "rcc\_object3d\_lisp\_ir"  $p_1 p_2 \dots p_n$  ""  $\varphi f_{yk}$ )

- Description:

The result of calling the function is inserting a rebar in the given coordinates, with a given diameter of the rebar's cross-section and the values of the steel's yield point.

- Parameters:

$P_1-P_1$  - coordinates of the characteristic points of the rebar. A single point is saved using a list containing three numbers defining the coordinates (x y z), e.g. (list 0 0 0),

$\varphi$  – diameter of the cross-section of the rebar



## Available features of the ArCADia BIM system

$f_{yk}$  – yield point of the rebar's steel .

- A function example:

```
(command "rcc_object3d_lisp_ir" (list 0 0 50) (list 0 0 0) (list 200 0 0) (list 0 0 50)
"" 1.2 435)
```

- A script example:

An example of the source code - inserting a rebar:

```
;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_lisp_ir" (list 0 0 50) (list 0 0 0) (list
200 0 0) (list 200 0 50) "" 1.2 435)
(command "rcc_sm" "Algorithm made correctly." 0)
```

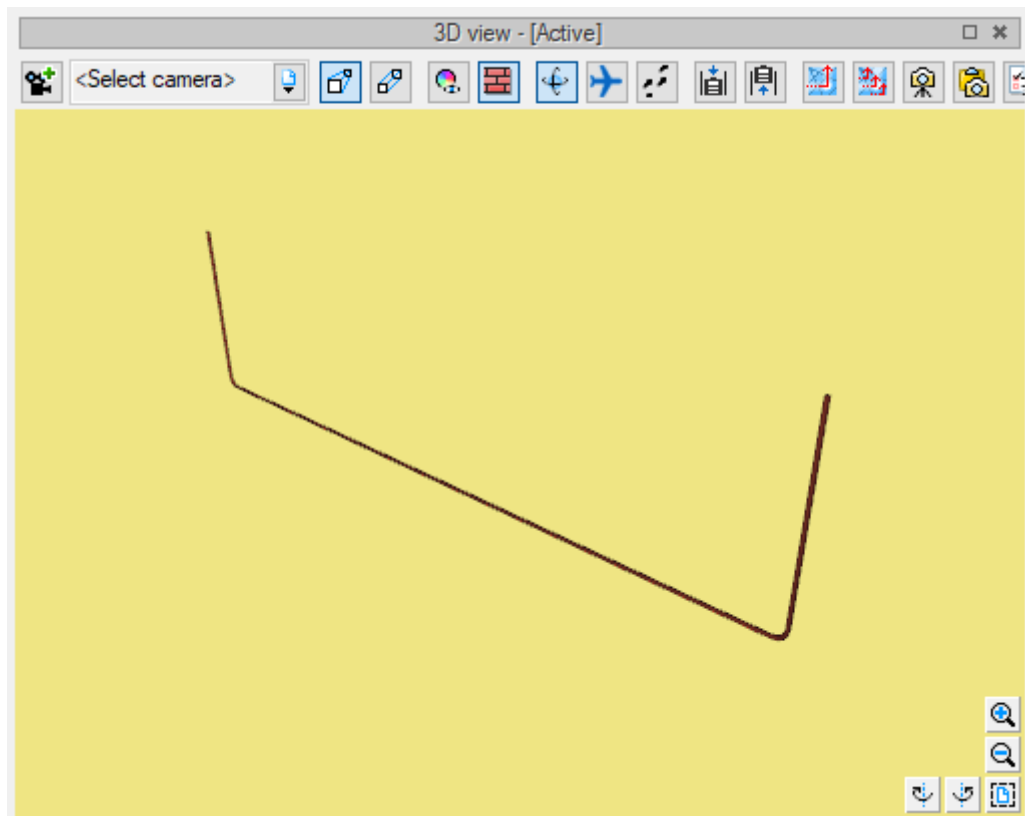


Fig. 42 3D view of the rebar

### 5.5.2. The stirrups

- Command:

(**command** "rcc\_object3d\_lisp\_ils"  $t_1$  (*list*  $x_1$   $y_1$   $z_1$ )

(*list*  $x_2$   $y_2$   $z_2$ ) (*list*  $x_3$   $y_3$   $z_3$ )  $n$   $s$   $\varphi$   $f_{yk}$ )

- Description:

The result of calling the function is inserting stirrups with a defined type, in a specific location, spacing and with a given diameter of the rebar's cross-section.

- Parameters:

$t_1$ — the type of a stirrup. Accepted values: 0 – a two-cut stirrup, 1 – a four-cut stirrup, 2 - a round stirrup,

$n$  – amount

$s$  – spacing,

$\varphi$  – the diameter of the rebar's cross-section

$f_{yk}$  – the yield point of the rebar's steel

The two-cut and four-cut stirrups:

$x_1$   $y_1$   $z_1$  - the coordinates of a vector containing one of the stirrup's arms. Any two coordinates must be equal to 0, the remaining value must be equal to the size of the stirrup arm being considered.

$x_2$   $y_2$   $z_2$  - the coordinates of the vector containing the second stirrup arm. Any two coordinates must have a value of 0, the remaining value must be equal to the size of the second stirrup arm. The first and second vector cannot be collinear.

$x_3$   $y_3$   $z_3$  – the coordinates of the stirrup insertion point.

The round stirrups:

$x_1$   $y_1$   $z_1$  – the coordinates of the vector containing the stirrup. Any two coordinates must have a value of 0, the remaining value must be equal to the size of the stirrup diameter.

$x_2$   $y_2$   $z_2$  – the coordinates of the vector containing the stirrup. Any two coordinates must have a value of 0, the remaining value must be equal to the size of the stirrup diameter. The first and second vector cannot be collinear.

$x_3$   $y_3$   $z_3$  – the coordinates of the stirrup insertion point.

- A function example:

(**command** "rcc\_object3d\_lisp\_ils" 0 (*list* 0 0 40) (*list* 40 0 0)

(*list* 10 20 30) 5 20 0.8 500)

- A script example:

An example of the source code - inserting a set of stirrups:

```
; #xml#<dialog>
```

## Available features of the ArCADia BIM system

```

;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_lisp_ils" 0 (list 0 0 40) (list 40 0 0)
(list
10 20 30) 5 20 0.8 500)
(command "rcc_sm" "Algorithm made correctly." 0)

```

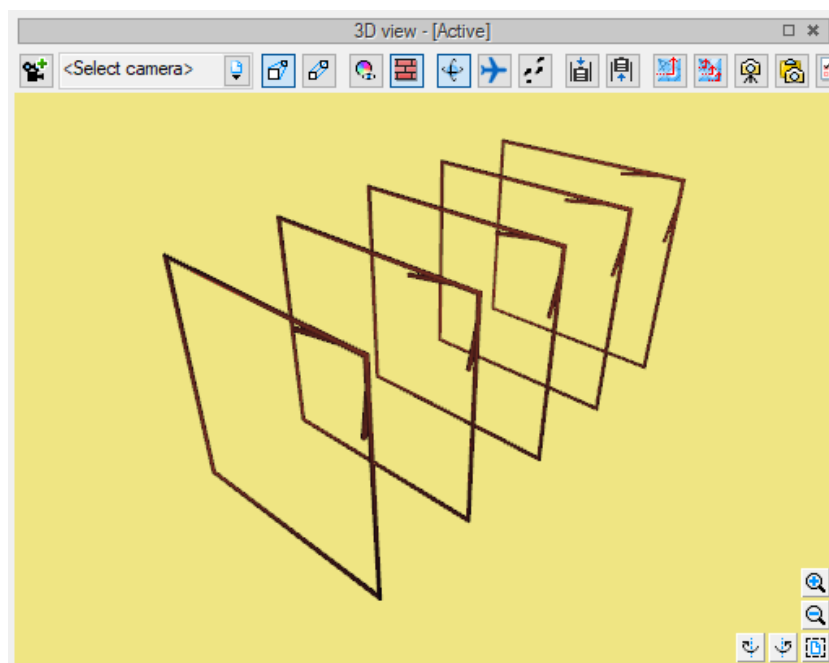


Fig. 43 Widok 3D zestawu strzemion A 3D view of the stirrup set

## 5.6. Inserting add-ons

### 5.6.1. Szczegóły wszystkich prętów Details of all rebars

- Command:  
(**command** "rcc\_object3d\_iard" (*list* *x y z*))
- Description:  
The result of calling the function is inserting details of all rebars.
- Parameters:

## Available features of the ArCADia BIM system

x y z— the coordinates of the insertion point of the reinforcement avatars.

- A function example:

(**command** "rcc\_object3d\_iard" (list 0 0 0))

- A script example:

An example of the source code - inserting reinforcement avatars:

```
;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 200 200 50)
(command "rcc_object3d_isv_north" (list 200 0 0))
(command "rcc_object3d_lisp_ir" (list 5 5 5) (list 195 5 5) ""
1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 195 5) (list 195 195 5)
"" 1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 5 45) (list 195 5 45) ""
1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 195 45) (list 195 195 45)
"" 1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 5 5) (list 195 195 45) ""
1.2 435)
(command "rcc_object3d_iard" (list 400 -100 0))
(command "rcc_sm" "Algorithm made correctly." 0)
```

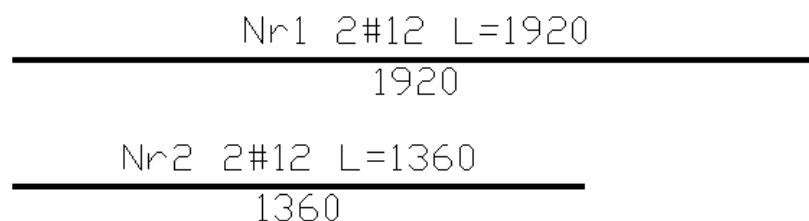


Fig. 44 3D view of the stirrup set

### 5.6.2. Details of the rebars from the view - horizontally

- Command:

(**command** "rcc\_object3d\_iardfvh" (*list x y z*))

- Description:

The result of calling the function is inserting from the active view details of rebars coinciding with the plane of the active view, which are horizontally distributed.

- Parameters:

**x y z**— the coordinates of the reinforcement avatars insertion point.

- A function example:

(**command** "rcc\_object3d\_iardfvh" (list 0 0 0))

- A script example:

An example of the source code - inserting reinforcement avatars:

```
;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 200 200 50)
(command "rcc_object3d_isv_north" (list 200 0 0))
(command "rcc_object3d_lisp_ir" (list 5 5 5) (list 195 5 5) ""
1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 195 5) (list 195 195 5)
"" 1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 5 45) (list 195 5 45) ""
1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 195 45) (list 195 195 45)
"" 1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 5 5) (list 195 195 45) ""
1.2 435)
(command "rcc_object3d_iardfvh" (list 400 -100 0))
(command "rcc_sm" "Algorithm made correctly." 0)
```

### 5.6.3. Details of the rebars from the view - vertically

- Command:

(**command** "rcc\_object3d\_iardfvv" (*list x y z*))

- Description:

The result of calling the function is inserting from the active view details of rebars coinciding with the plane of the active view, which are vertically distributed.

- Parameters:

**x y z**— the coordinates of the reinforcement avatars insertion point.

- A function example:

(**command** "rcc\_object3d\_iardfvv" (list 0 0 0))

- An script example:

An example of the source code - inserting reinforcement avatars:

```
;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 200 200 50)
(command "rcc_object3d_isv_north" (list 200 0 0))
(command "rcc_object3d_lisp_ir" (list 5 5 5) (list 195 5 5) ""
1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 195 5) (list 195 195 5)
"" 1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 5 45) (list 195 5 45) ""
1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 195 45) (list 195 195 45)
"" 1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 5 5) (list 195 195 45) ""
1.2 435)
(command "rcc_object3d_iardfvv" (list 400 -100 0))
(command "rcc_sm" "Algorithm made correctly." 0)
```

#### 5.6.4. Missing details of the rebars

- Command:

(**command** "rcc\_object3d\_iird" (*list x y z*))

- Description:

The result of calling the function is inserting missing details of reinforcing bars.

- Parameters:

**x y z**— the coordinates of the reinforcement avatars insertion point.

- A function example:

(**command** "rcc\_object3d\_iird" (list 0 0 0))

- A script example:

An example of the source code - inserting reinforcement avatars:

```
;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 200 200 50)
(command "rcc_object3d_isv_north" (list 200 0 0))
(command "rcc_object3d_lisp_ir" (list 5 5 5) (list 195 5 5) ""
1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 195 5) (list 195 195 5)
"" 1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 5 45) (list 195 5 45) ""
1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 195 45) (list 195 195 45)
"" 1.2 435)
(command "rcc_object3d_lisp_ir" (list 5 5 5) (list 195 195 45) ""
1.2 435)
(command "rcc_object3d_iardfvh" (list 400 -100 0))
(command "rcc_object3d_iird" (list 450 -50 0))
(command "rcc_sm" "Algorithm made correctly." 0)
```

### 5.6.5. A steel list of all elements

- Command:

(**command** "rcc\_object3d\_isl" (*list x y z*))

- Description:

The result of calling the function is inserting of a steel lists in the given location for all elements, i.e. all reinforced concrete elements defined in the project.

- Parameters:

*x y z* – the coordinates of the reinforcement avatars insertion point.

- A function example:

(**command** "rcc\_object3d\_isl" (list 0 0 0))

- A script example:

An example of the source code – the steel lists of all elements:

```
;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 50 50 200)
(command "rcc_object3d_isv_north" (list 200 0 0))
(command "rcc_object3d_lisp_ir" (list 4 4 4) (list 4 4 196) ""
1.2 435)
(command "rcc_object3d_lisp_ir" (list 46 4 4) (list 46 4 140) ""
1.2 435)
(command "rcc_object3d_lisp_ir" (list 46 46 4) (list 46 46 140)
"" 1.2 435)
(command "rcc_object3d_lisp_ir" (list 4 46 4) (list 4 46 196) ""
1.2 435)
(command "rcc_object3d_isl" (list 500 -100 0))
(command "rcc_sm" "Algorithm made correctly." 0)
```



## Available features of the ArCADia BIM system

Steel list

No.	Quantity [pcs]	Diameter [mm]	Single length [mm]	Total length [m]
				435.0 MPa
				#12.0
1	2	12.0	1920	3.84
2	2	12.0	1360	2.72
Total length [m]				6.6
Unit weight [kh/m]				0.888
Weight [kg]				5.8
Total weight [kg]				5.8

Construction concrete C30/37

Fig. 45 The steel lists table inserted in the working area

## 5.6.6. A steel list of a single element

- Command:

(**command** "rcc\_object3d\_islb" (*list x y z*))

- Description:

The result of calling the function is inserting of a steel lists in the given location for a single element.

- Parameters:

*x y z* – the coordinates of the reinforcement avatars insertion point.

- A function example:

(**command** "rcc\_object3d\_islb" (list 0 0 0))

- A script example:

An example of the source code – a single element

```
;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
(command "rcc_object3d_io" "Reinforced concrete component No. 1"
30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 50 50 200)
(command "rcc_object3d_isv_north" (list 200 0 0))
(command "rcc_object3d_lisp_ir" (list 4 4 4) (list 4 4 196) ""
1.2 435)
(command "rcc_object3d_lisp_ir" (list 46 4 4) (list 46 4 140) ""
1.2 435)
```

## Available features of the ArCADia BIM system

```
(command "rcc_object3d_lisp_ir" (list 46 46 4) (list 46 46 140)
"" 1.2 435)

(command "rcc_object3d_lisp_ir" (list 4 46 4) (list 4 46 196) ""
1.2 435)

(command "rcc_object3d_islb" (list 500 -100 0))

(command "rcc_sm" "Algorithm made correctly." 0)
```

Steel list - Reinforced concrete component No. 1

No.	Quantity [pcs]	Diameter [mm]	Single length [mm]	Total length [m]
				435.0 MPa
				#12.0
1	2	12.0	1920	3.84
2	2	12.0	1360	2.72
Total length [m]				6.6
Unit weight [kh/m]				0.888
Weight [kg]				5.8
Total weight [kg]				5.8

Construction concrete C30/37

Fig. 46 A steel list table inserted in the working area

## 5.6.7. The special characters in the text of controls

In order to obtain special characters in the text of labels, the appropriate formula should be used:

"&#" + Kod + ";"

For example Ω: &#x03A9;

Symbol	Code
©	x00A9
°	x00B0
±	x00B1
²	x00B2
³	x00B3
⁵	x2075
·	x00B7
/	x2044
→	x2192

## Available features of the ArCADia BIM system


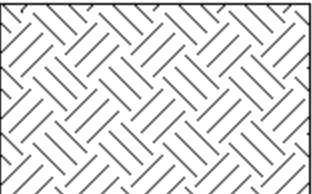
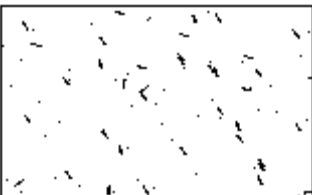
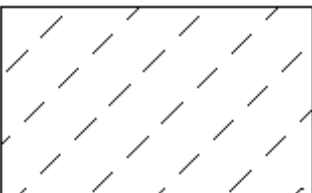
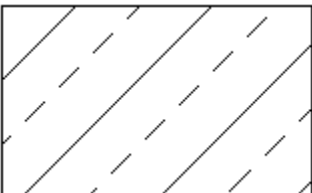
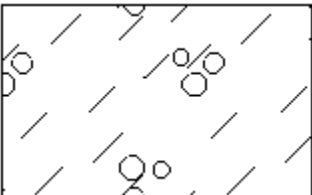
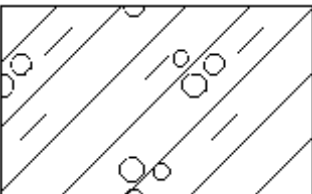
≠	x2260
∞	x221E
≈	x2248
≤	x2264
≥	x2265
A	x0391
B	x0392
Г	x0393
Δ	x0394
E	x0395
Z	x0396
H	x0397
Θ	x0398
I	x0399
K	x039A
Λ	x039B
M	x039C
N	x039D
≡	x039E
O	x039F
Π	x03A0
P	x03A1
Σ	x03A3
T	x03A4
Υ	x03A5
Φ	x03A6
X	x03A7

## Available features of the ArCADia BIM system

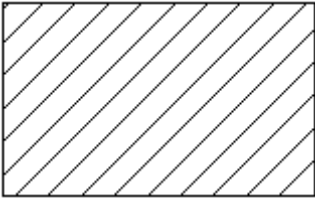
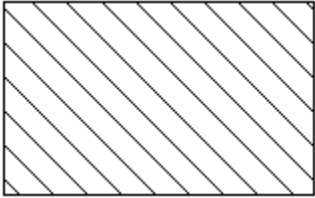
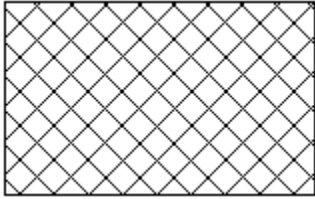
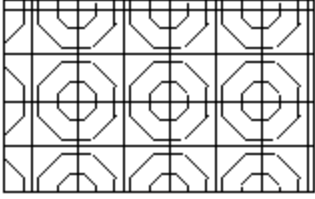
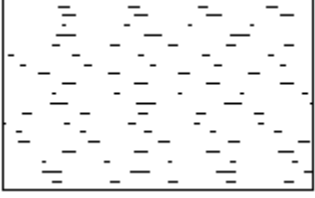

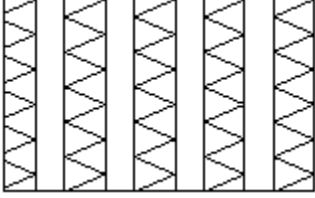
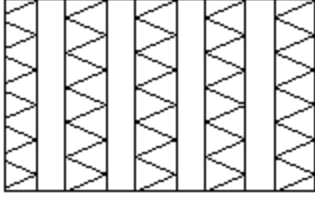
$\Psi$	x03A8
$\Omega$	x03A9
$\alpha$	x03B1
$\beta$	x03B2
$\gamma$	x03B3
$\delta$	x03B4
$\varepsilon$	x03B5
$\zeta$	x03B6
$\eta$	x03B7
$\theta$	x03B8
$\iota$	x03B9
$\kappa$	x03BA
$\lambda$	x03BB
$\mu$	x03BC
$\nu$	x03BD
$\xi$	x03BE
$\omicron$	x03BF
$\pi$	x03C0
$\rho$	x03C1
$\sigma$	x03C3
$\tau$	x03C4
$\upsilon$	x03C5
$\phi$	x03C6
$\chi$	x03C7
$\psi$	x03C8
$\omega$	x03C9

### 5.6.8. Hatching


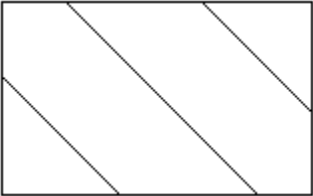
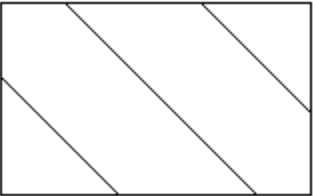
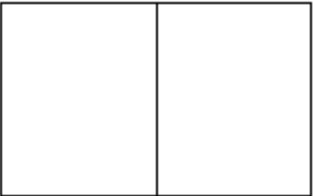
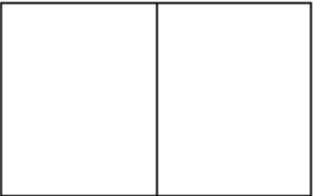
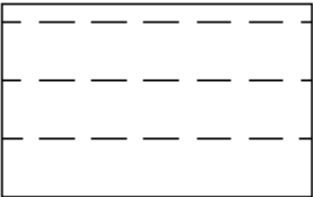
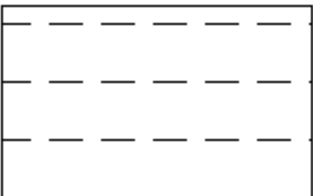
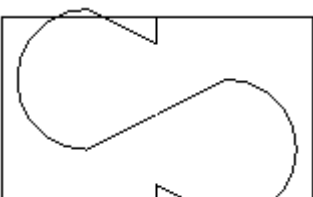
The table shows the hatching patterns of surfaces lying on the plane.

Value	Hatch pattern
20000	
20001	
20002	
20003	
20004	
20005	
20006	

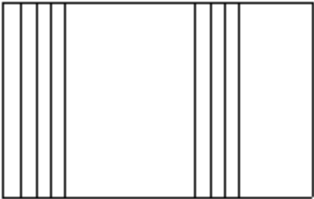

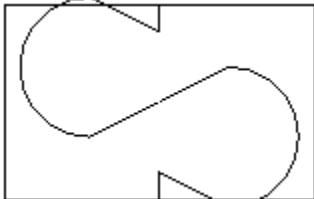
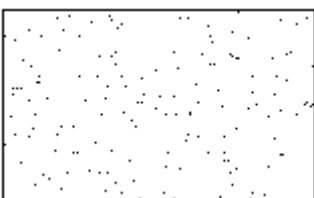
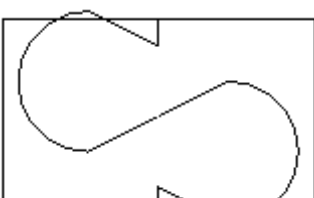
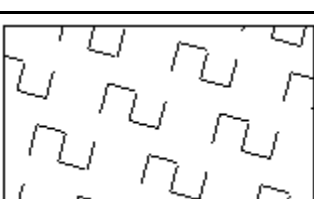

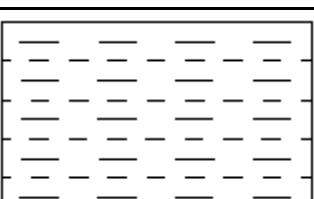
## Available features of the ArCADia BIM system

20007	
20008	
20009	
20010	
20011	
20012	
20013	
20014	

## Available features of the ArCADia BIM system

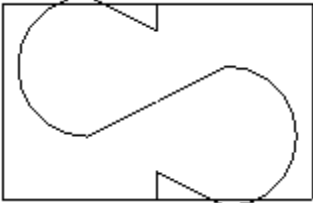
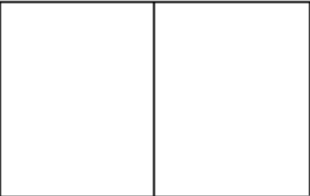
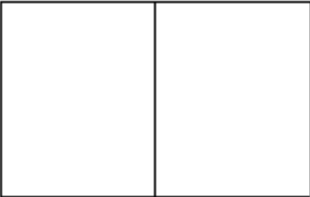
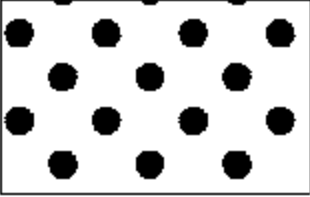

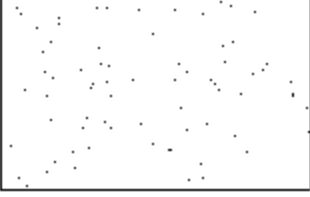

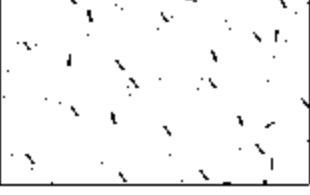
20015	
20016	
20017	
20018	
20019	
20020	
20021	
20022	

## Available features of the ArCADia BIM system

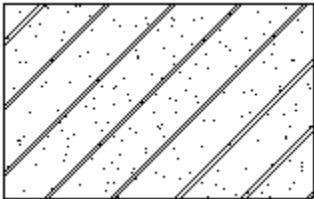
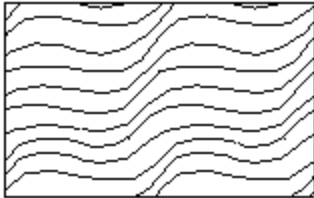
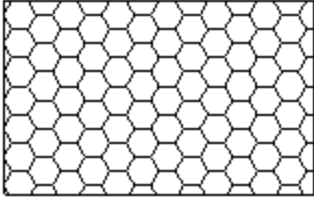
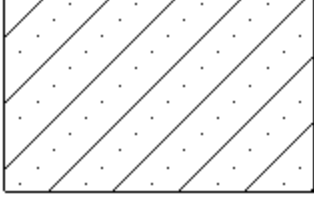
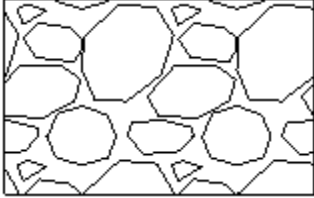
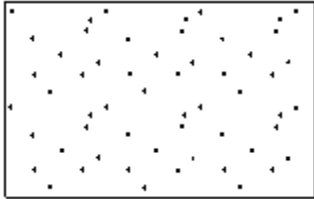
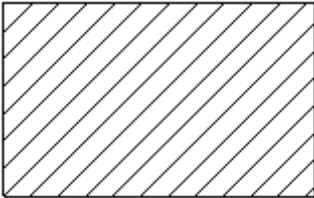
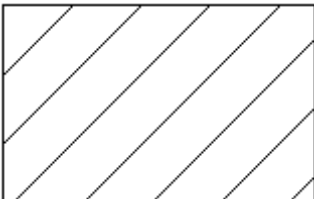
20023	
20024	
20025	
20026	
20027	
20028	
20029	
20030	



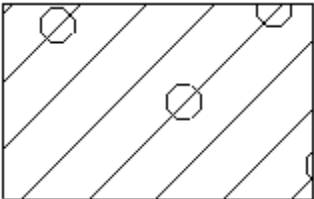
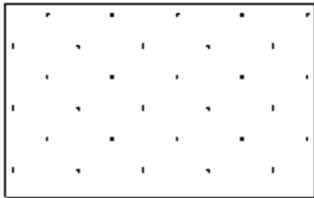
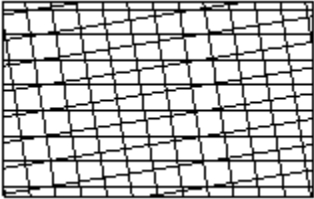
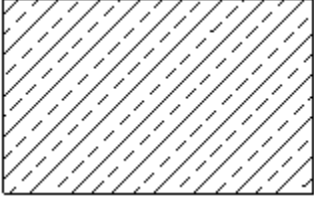

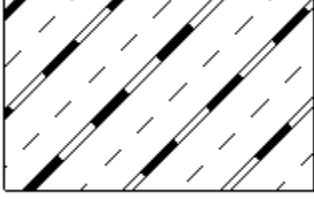
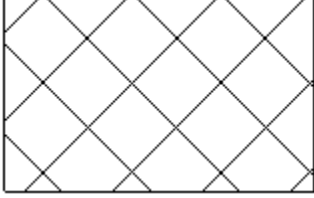
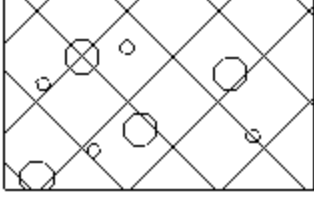
## Available features of the ArCADia BIM system

20031	
20032	
20033	
20034	
20035	
20036	
20037	
20038	

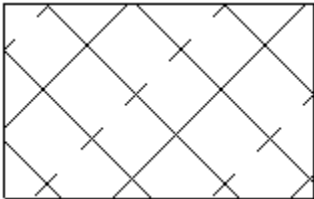
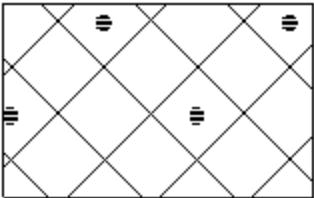
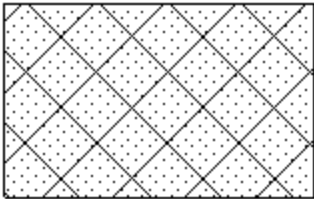
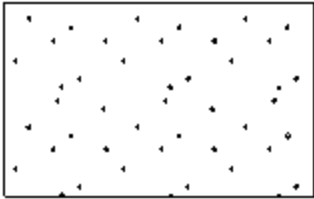
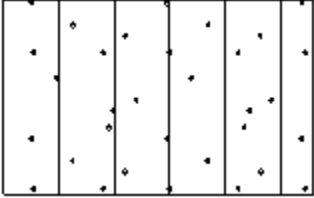
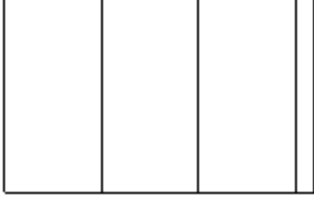
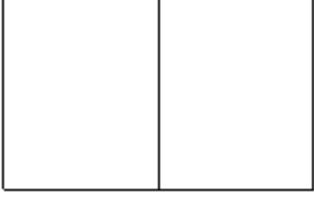
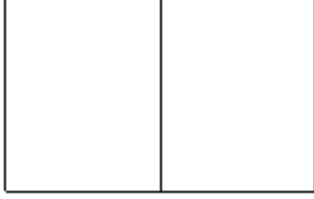
## Available features of the ArCADia BIM system

20039	
20040	
20041	
20042	
20043	
20044	
20045	
20046	

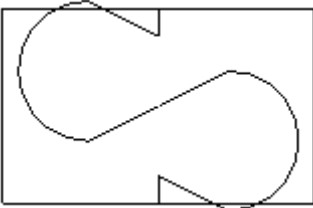
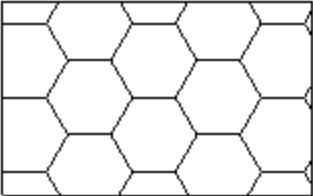
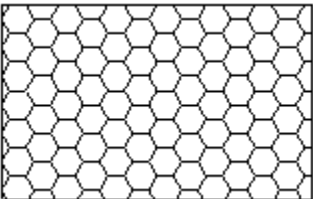
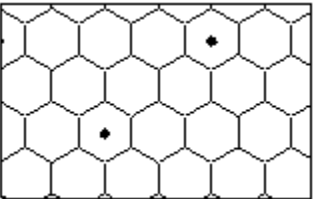
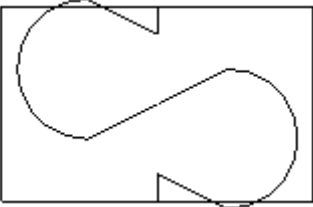
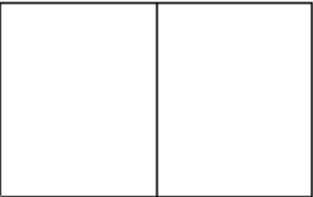
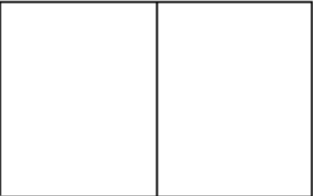

## Available features of the ArCADia BIM system

20047	
20048	
20049	
20050	
20051	
20052	
20053	
20054	


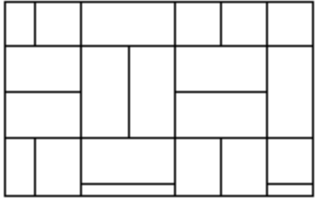
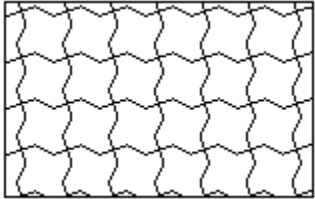
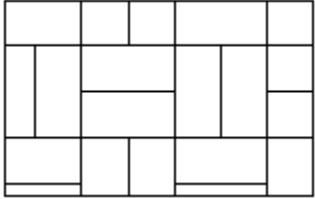
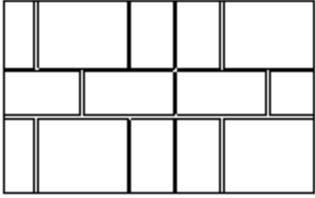
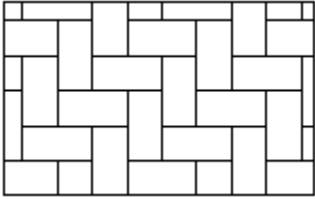
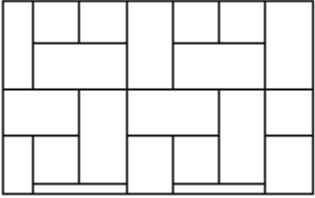
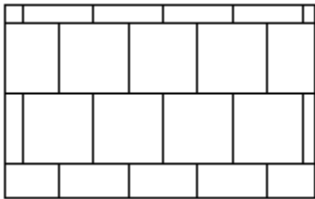
## Available features of the ArCADia BIM system

20055	
20056	
20057	
20058	
20059	
20060	
20061	
20062	

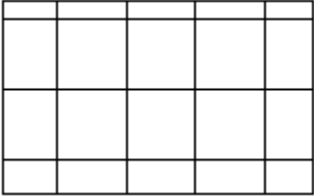
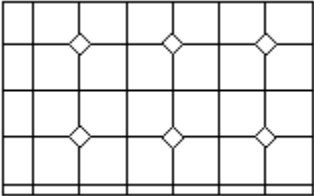
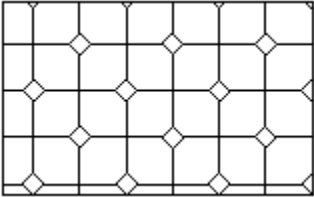
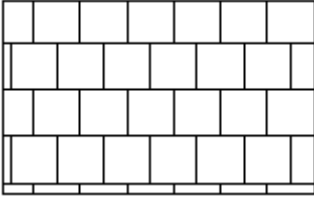
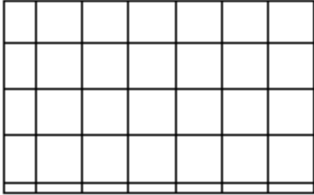
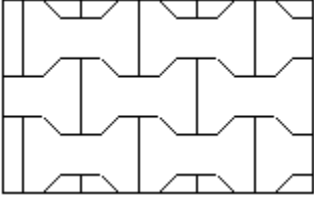
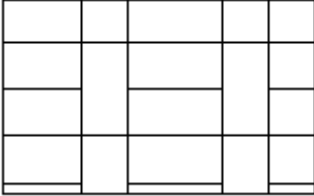
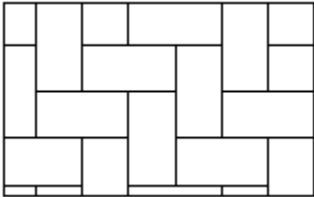
## Available features of the ArCADia BIM system

20063	
20064	
20065	
20066	
20067	
20068	
20069	
20070	

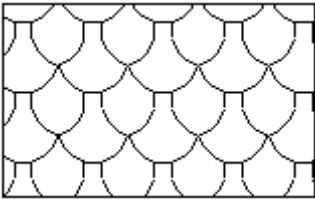
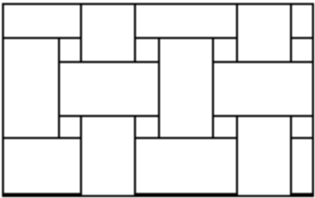
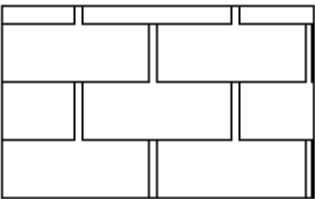
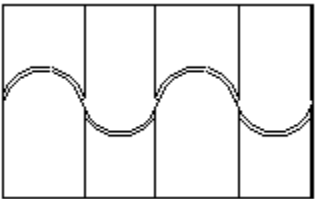
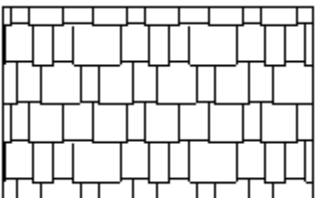
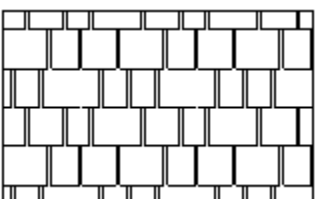
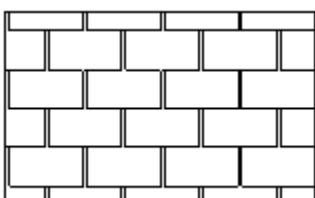
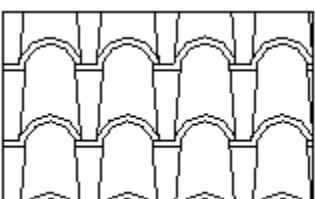
## Available features of the ArCADia BIM system

20071	
20072	
20073	
20074	
20075	
20076	
20077	
20078	

## Available features of the ArCADia BIM system

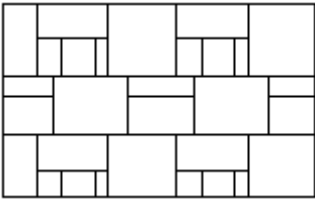
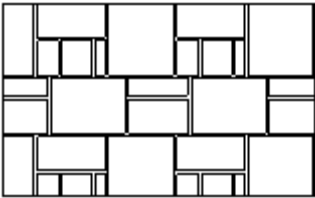
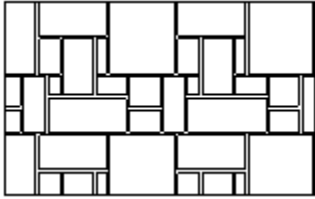
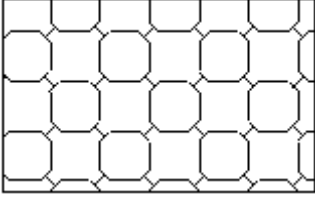
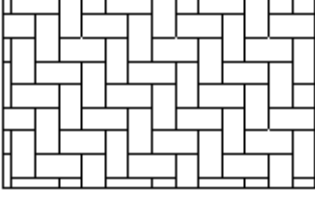
20079	
20080	
20081	
20082	
20083	
20084	
20085	
20086	

## Available features of the ArCADia BIM system

20087	
20088	
20089	
20090	
20091	
20092	
20093	
20094	



## Available features of the ArCADia BIM system

20095	
20096	
20097	
20098	
20099	

## 5.7. The monitoring of script execution progress

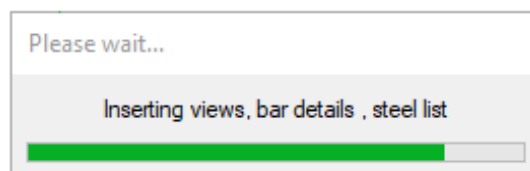


Fig. 47 A sample of a progress bar

- Command:  
(**command** "rcc\_object3d\_lisp\_swd" nazwa)
- Description:  
The result of calling the function is running the progress window and displaying the specified text in it.

## Available features of the ArCADia BIM system

- Parameters:

*name* - the content of the displayed text.

- A function example:

(**command** **rcc\_object3d\_lisp\_swd** "Inserting the object solid")

- Command:

(**command** "rcc\_object3d\_lisp\_wdpp" *x*)

- Description:

The function allows you to move the value of progress by a given value. The sum of the values given for all invoked commands cannot exceed 100.

- Parameters:

*x* – the value of shifting the progress of the process.

- A function example:

(**command** "rcc\_object3d\_lisp\_wdpp" 25)

- Command:

(**command** "rcc\_object3d\_lisp\_wdmsg" *nazwa*)

- Description:

The command changes the text content in the progress bar

- Parameters:

*name* - the content of the displayed text.

- A function example:

(**command** "rcc\_object3d\_lisp\_wdmsg" "Inserting reinforcement")

- Command:

(**command** "rcc\_object3d\_lisp\_wdfp")

- Description:

The command moves the progress value to the full execution state from the currently displayed value bar.

- Parameters:

—

- Command:

(**command** "rcc\_object3d\_lisp\_hwd")

**Available features of the ArCADia BIM system**

- Description:

It hides the progress window.

- Parameters:

—

- A script sample:

An example of a script that uses the given commands:

```
;#xml#<dialog>
;#xml# <panels/>
;#xml#</dialog>
;#lispParams#
;#lispParams#
;(command "rcc_object3d_lisp_swd" "Reinforced concrete component
No. 1")
(command "rcc_object3d_io" "Projekt" 30 2000)
(command "rcc_object3d_aco" (list 0 0 0) 50 50 200)
(command "rcc_object3d_lisp_swd"
"$_(LispScripts.BeamWithPlate.message_1)")
(command "rcc_object3d_lisp_wdpp" 100)
(command "rcc_object3d_lisp_wdmsg" "Inserting reinforcement")
(command "rcc_object3d_lisp_ir" (list 4 4 4) (list 4 4 196) ""
1.2 435)
(command "rcc_object3d_lisp_wdpp" 10)
(command "rcc_object3d_lisp_ir" (list 46 4 4) (list 46 4 140) ""
1.2 435)
(command "rcc_object3d_lisp_wdpp" 10)
(command "rcc_object3d_lisp_ir" (list 46 46 4) (list 46 46 140)
"" 1.2 435)
(command "rcc_object3d_lisp_wdpp" 10)
(command "rcc_object3d_lisp_ir" (list 4 46 4) (list 4 46 196) ""
1.2 435)
(command "rcc_object3d_lisp_wdpp" 10)
(command "rcc_object3d_lisp_wdmsg" "Inserting cross-section")
(command "rcc_object3d_lisp_wdfp")
(command "rcc_object3d_islb" (list 500 -100 0))
(command "rcc_object3d_lisp_hwd")
(command "rcc_object3d_isv_north" (list 200 0 0))
(command "rcc_sm" "Algorithm made correctly." 0)
```