2.1.0

US 781	cee::ug::ElementHighligter: Added edge lines to element highlight.
US 765	Optimized display model generation (UnstructGridModel::updateVisualization()). Updates that only require display model modification (changing draw styles etc) are at least 2x faster on Multi Core machines. All updates are faster utilizing multi-core computations.
US 764	Qt: Added support for using ANGLE/MESA renderers (e.g. Remote Desktop) All calls to OpenGL in C3DC are now wrapped and can be implemented by different back-ends. For Qt, we provide a ready to use back-end (cee::qt::OpenGLFunctionsBackend_qt5) that forwards all OpenGL calls to Qt's implementation. This allows for use of ANGLE or MESA OpenGL backings, which make it possible to run the app in an environment without hardware accelerated OpenGL (e.g. remote desktop). To use the back-end with Qt, use the following method to create the context group: contextGroup = VisualizationComponent::createOpenGLContextGroup(new cee::qt::OpenGLFunctionsBackend_qt5); After plugging in the back-end, you can specify Qt's application attributes to force the choice of renderer, e.g: QApplication::setAttribute(Qt::AA_UseSoftwareOpenGL);
US 763	Optimization of isovolume calculations. General speedup around 1.6x. In some cases (high-order surface elements) 10x+ faster.
US 762	Optimized mode shape animation. Mode shape animations can now be run with interpolation of position and (optionally) scalar result on the GPU, thus vastly reducing the memory usage and the setup time. This option is enabled by default, but can be disabled to get the old behavior with the ModelSettings::setUseShaderBasedModeShapeAnimation(false) method.
US 757	Added rayIntersectAllHits to UnstructGridModel. This method returns all items in the model that are hit by a given ray, not only the first one as rayIntersect does. The returned HitItemCollection can then be processed. It is uesful for example, to capture overlapping elements or filtering out elements of different types from those hit by the ray.
US 756	Add models to a final rendering pass with clearing of depth buffer. Using View::addSecondPassModel() you can add a model to the view that will be rendered in a separate pass after all other models. The depth buffer is cleared before this pass, so all models added with this method will be rendered on top of the normal models.
US 747	We have listened to our customers. The linear combination of states was taking too much memory (and hence too much time). DataSourceLinearCombination has been removed from the distribution, and is replace by StateLinearCombination, much leaner and much faster. The class documentation explains the new usage.
US 744	Added support for picking (rayIntersect) for MarkupPartArrows and MarkupPartLines
US 675	Added support for showing undeformed model. When showing a model with displacements, the undeformed model can now be shown as either lines (mesh) or outline. The line color can be set to a single color or follow the part color.
US 662	Added query methods to get source element info from a cutting plane surface. See CuttingPlaneData allTrianglesSourceElementIndices() and allTrianglesSourcePartIndices() for more info.
US 655	Added support for color coding UG models based on DataElementSets and Element Type. Introduced ModelColorSource in ModelSpec. Use ModelSpec::setModelColorSource() to specify how to color the model when no fringes result is used. Possible options: - PART: default, as before - SET: Color code the model by DataElementSet. Specify the set color with UnstructGridModel::setElementSetColor(). - ELEMENT_TYPE: Color code each element based on the element type. If SET or ELEMENT_TYPE, a category legend describing each color used will be shown.

2.0.0

US 737	Added ElementQuery::mapScalarResult(). Renamed mapResults() to mapScalarResults()
US 731	Added support for instanced geometry with 'Fixed Pixel Size' support in the Markup Model. Added MarkupPartInstancedGeometry, which allows for instancing of a single geometry just by specifying a transformation matrix for each instance. The class supports setting a color per instance. The performance is a lot better than using MarkupPartTriangles or the geometry model for each part.
US 730	ResultInfo supports a number of attributes, in the form of (name, value) pairs of strings
US 725	Added support for showing all vectors. This will show all vectors, also the interior ones (not referenced by the surface of the model). Added vector setting "setShowInteriorVectors".
US 674	Isovolumes now also support surface (triangles, quads, etc) elements. This allows for using isovolumes for sub-element filtering. Scalar filtering is done on an element level, where the element it either visible or not. With isovolumes, element that are partly within range are split up, creating the correct filtering.
US 673	Element highlighter label includes scalar value
US 671	Added support for polyhedron elements in the UnstructGrid Component. Added support for polyhedron elements in VTFx. OpenFOAM and VTU readers now supports polyhedrons.
US 670	Added VisualizationPartQuery.visibleElementSurfaces() which returns all the visible element surfaces of a given part.
US 669	ResultInfo now provides the nature of a result (as in physical nature). This allows consistent identification of results, regardles of their name
US 664	Added support for using relative values when exporting a displacement result to VTFx.
US 663	Reflective mirroring is now available in MirrorSettings.
US 594	Introduced new class ResultsQuery. This class provides functionality to retrieve results on selected nodes and elements from a given data source. In the future versions, it will also be the place to perform postprocessing on the results found in the data source.
US 426	Added support for import/export of element sets. Any sets present in the CAE file opened with the ImportCAE component will be available for use in the UnstructGrid Component.

1.4.9

US 652	Added filtering of element categories in Vdm reader. Added a setting to ReaderSettings in DataSourceCae that allows to specify element categories that should be ignored when reading the model. Element categories refere to the FEM nature of elements. They are defined in enum cee::ug:Element::Category. To exclude an element category from the reader, call dataSource->readerSettings().excludeElementCategory(category) prior to opening the CAE data source. To clear the exclusion mask : dataSource->readerSettings().clearExcludedElementCategories()
US 651	Added ExportDataSourceVTFx to Export component. This class allows to save all data (all states / all results) from a data source to a VTFx-formatted file.
US 650	Added support for specifying title and tick mark fonts in the cee::vis::OverlayColorLegendContinuousDomain Added font to constructor, and added methods for setting title and tick mark fonts.
US 648	Added new TrueTypeFont class to be used instead of cee::vis::Font::createTrueTypeFont(...), which is now deprecated. The new class supports changing the font size on the fly. Any associated overlay classes may need to update it's size to reflect the new font size, e.g. by calling OverlayTextBox::setSizeToFitText(). MarkupPartLabels will be updated automatically.
US 647	Refactored Python interface. Based on feedback from our users, we have suppressed attributes in the Python interface in order to stick more strictly to the C++ interface. This gets rid of a number of ambiguities that arose using the initial distribution. Starting with C3DC 1.4.9, the Python interface now offers the standard set'ers and get'ers found in the C++ distribution and documentation.

1.4.8

US 634	Added option to specify a color to use if the color legend has a zero range (min == max). The color legend will then be drawn with one large area with this color and the corresponding value in the middle. See ScalarSettings::setSingleColorZeroRangeMode() for more info.
US 633	Added more frequent tests to user abort during updateVisualization() to get a more responsive cancelling of the operation
US 632	Added option to disable the automatic removal of overlapping tick mark labels on the color legend. Set cee::ug::ScalarSettings::setSkipOverlappingTickMarkLabels(false) to disable the automatic hiding.

1.4.7

US 618	Optimized display model generation in the unstruct grid model (cee::ug::UnstructGridModel::updateVisualization()). Models with flat shading are much faster now. Also a decent increase on smooth shaded models.
US 616	Added DataSourceStateLinearCombination, a DataSourceMemory used to generate a linear combination of states from a given UnstructGridModel
US 615	Added left multiplication by scalar to Vec3d
US 614	Added a base result type and id, as well as a derive operation to ResultInfo. This allows to provide methods in DataSourceMemory to compute results that derive from others, such as components from vectors or principal values from tensors
US 613	SymmetricTensor : added left multiplication by scalar, multiplication by vector, trace, deviator tensor, a static method to get the identity tensor, vonMises computation, and principal (eigen) value and vector computations
US 610	Added option to control the maximum height of the unstruct grid color legend when using automatic layout (cee::vis::Overlay::setMaximumHeightAutomaticLayout()).
US 601	Added reader setting handling to DataSourceCae. Implemented "Use Mentat Linear Extrapolation" setting on MARC reader
US 599	Added support for rendering a reversed color legend (cee::ug::ScalarSettings::setDrawLegendWithMinimumValueOnTop()).
US 597	Added DataSourcePhaseResponse, a DataSourceMemory that provides the phase-response of a modal analysis at a given frequency.
US 596	Added new ElementsQuery to get the volume of an element. Returns 0 for non-volume elements. The computation is based on linear polyhedra, hence the volume for quadratic elements is an approximation
US 595	Added a reader for PTC neutral files. The reader loads one analysis of a PTC design solution. It expects a PTC neutral file (.neu) as entry

1.4.6

US 593	Changed ScalarSettings::legendVisible(bool) to ScalarSettings::legendVisibilityMode(). Added support for setting the color legend to auto (show the legend if used by any item in the 3d view, default), always (show the legend independent of use) or never (always hide). To port old code, modify setLegendVisible(true) to setLegendVisibilityMode(ScalarSettings::AUTO) and setLegendVisible(false) to setLegendVisibilityMode(ScalarSettings::NEVER).
US 592	Added option for always using polygon offset for parts in UnstructGrid model. This might be useful if you have parts with primitives in the same plane and are using eye-lift (setEyeLiftFactor) in order to control the visibility of the part. Added cee::ug::PartSettings::setAlwaysUsePolygonOffset()
US 591	Increased the number of clipping planes in one View from 6 to 12. This applies both to cutting plane clipping (cee::ug::CuttingPlane::setClipping()) and to View clipping (cee::vis::Clipping).
US 587	Improved accuracy of filled contours scalar mapping when having (very) uneven levels (cee::ug::ColorMapper::type() == FILLED_CONTOURS).
US 585	Added new wheelEvent() for CameraInputHandler for enabling zoom to cursor functionallity
US 581	Add reader for MSC/Marc .t16 and .t19 files in the ImportCae component.
US 574	Added support for undefined, above and below range colors to the scalar mappers deriving from ScalarMapperContinuousDomain. Added EffectTexture::createResultMapping() to help setup a texture effect suited for results mapping. More accurate results mapping in the ScalarMapperFilledContours when the levels differ vastly in size.
US 572	Removed ScalarSettings::enableFringesElementFiltering(). Filtering is enabled when calling setFringesElementFilteringVisibleRange() and disabled by calling disableFringesElementFiltering().

US 564	Added cee::ug::DataElementSetGenerator::createAllElementsPart()

1.4.5

US 562	GeometryModel::setModelTransformation and modelTransformation renamed to setTransformation() and transformation(). Signature also changed to return a const ref (in c++).
US 561	Extension and refactor of scalar mappers and color legend and in cee::Visualization. Added 4 new scalar mappers: ScalarMapperFilledContoursUniform, ScalarMapperFilledContours, ScalarMapperContinuous and ScalarMapperContinuousPiecewise. Added OverlayColorLegendContinuousDomain which supports all four mappers. Changes: ScalarMapperUniformLevels is replaced by ScalarMapperFilledContoursUniform. OverlayColorLegend is replaced by OverlayColorLegendContinuousDomain. See documentation of cee::vis::OverlayColorLegendContinuousDomain for more information and examples.
US 552	Added cee::ug::ColorMapper::setTextureSize() to control the size of the texture used in the UnstructGridModel to show scalars.
US 458	Added support for drawing cutting planes with the color of the source part (if not mapping results onto the plane). Added CuttingPlane::setUseSourcePartColor().

1.4.4

US 507	Updated ImportCAE file readers with support for Abaqus 2016 model files
US 506	Updated ImportCAE file readers with bug fixes
US 504	Added readers for LS-DYNA files in the ImportCAE component. One reader handles input (keyword, *.k) files and an another one handles results (state database, "*.d3plot; *.ptf") files.
US 481	Added support for node sets in the UnstructGrid component. Added the DataNodeSet and DataNodeSetItem classes to store a node set. Added DataNodeSetGenerator class for generating node sets based on a region of the screen (rectangle or polygon), a plane or all the nodes in a given frame.
US 478	Added methods to access the underlying raw data pointer in part result classes (DataPartScalar, Vector, SymmetricTensor, Displacement, Visibility). This allows an optimized parsing of the values. It is the user's responsibility to ensure that parsing is within bounds.
US 477	Reduced VTFx export time when writing cases that use the same database
US 474	Added method for querying outline edges (based on the current crease angle setting) of an unstruct grid part. See VisualizationPartQuery::outlineEdges() for more info.
US 473	Added method for querying edges of an element in an unstruct grid part. See ElementsQuery::elementEdges() for more info.
US 472	Added multiple section definition support to Vdm-based readers. Certain Cae databases contain results that are based on more than one section definition. For example,stress and strain may on one and temperature on another.
US 468	Added a PartSetting for controlling the line width of the mesh and outline mesh lines. See PartSettings::setMeshLineWidth() for more info.
US 467	Added option to always show the labels on the axis cross (to disable the depth test vs the axis arrows). See OverlayAxisCross::setAlwaysShowLabels() for more info.
US 460	Added an STL reader to the ImportCAE component. Both ascii and binary files are supported. The reader produces a single-state, single-geometry, single-part model.
US 456	Added support for transformation matrices for all MarkupPart* classes.
US 427	Added frameIndex() to HitItem and PartHitItems in the UnstructGrid component.

1.4.3

US 464	Added text alignment to OverlayTextBox. Text alignment is set using cee::vis::OverlayTextBox::setTextContentAlignment(). Options are LEFT, CENTER or RIGHT.
US 463	Updated ImportCAE file readers with new features and bug fixes
US 461	Added helper method to ug::DataElements that returns an element's index given its id using a simple linear search. The method will assert if no element ids are defined.
US 457	Added DataElementSet::addItems(...). This is a much faster way to add multiple elements for a given part to a DataElementSet.
US 450	Added helper method to ug::DataNodes that returns a node's index given its id using a simple linear search. The method will assert if no node ids are defined.
US 447	Added a query to ElementsQuery to gather all elements that contain a given node
US 445	Added a query to ElementsQuery to gather all elements that contain an edge given by its two nodes
US 438	Added label size query (in pixels) to MarkupPartLabels. MarkupPartLabels::width() MarkupPartLabels::height() for more info.
US 437	Reduced vertex count in display model by 33% (QUAD/HEX) to 56% (QUAD8/HEX20) for models with per element and per element results.
US 436	Added View::regionClosestVertex(). This will find the vertex (in world coordinates) closest to the front clipping plane within the given screen based region.

1.4.2

US 425	Added support for cutting plane animation. Use the class cee::ug::CuttingPlaneAnimation to setup the animation.
US 423	Added support for uniform spacing of tickmarks on custom continuous color legends. ScalarSettings::setForceUniformLevelSpacing() have been renamed to setForceUniformTickMarkSpacing() and this now controls the color legend in both FILLED_CONTOURS mode and with custom continuous tick marks.
US 421	Added support for controlling scalar filtering of elements with undefined scalar results. Default is now that elements with a undefined scalar result is visible even when the scalar filtering is enabled. To hide elements with undefined scalar results, use the method ScalarSettings::setFilterUndefinedElements(true).
US 419	Improved performance for creating parts with OUTLINE and SURFACE_OUTLINE_MESH draw styles in the UnstructGridModel. Computation of outline and outline mesh lines is now 5-10 times faster. Switching the model from SURFACE to OUTLINE draw style is now 3-4 times faster.
US 418	Improved performance for generation and rendering of element mesh lines (SURFACE_MESH) in the UnstructGridModel. Computation of element mesh lines are now 5-10 times faster. Switching the model from e.g. SURFACE to SURFACE_MESH draw style is now 2-3 times faster.
US 400	Scalar Color mapper: Added support for setting opacity for values above and below the current range of the color legend. Added cee::ug::ColorMapper::setAboveRangeOpacity() and cee::ug::ColorMapper::setBelowRangeOpacity()

1.4.1

US 413	Improved drawing performance for the SURFACE_MESH draw style in the unstructured grid component
US 412	Added size_t View::modelIndex(Model* model) const
US 399	ug::ElementHighlighter: Added support for highlighting hitItems on cutting planes, isosurfaces and isovolumes. Added partId() to ug::HitItem which contains the id of the DataPart even for hitItems on cutting planes, isosurfaces and isovolumes.

1.4.0

US 369	Added a Python interface to Ceetron 3D Components. The interface is produced using SWIG and distributed in the form of a Python 3.4 or higher module named cee, with submodules core, ug, vis, geo, imp, exp, plt, and rep giving access to the functionality of each component of the C++ distribution : Core, UnstructGrid, Visualization, Geometry, ImportCAE, Export, Plot2d and Report. The Python interface is available for Windows only in this release. It has been compiled for 64-bit platforms and runs with Python 3.4 (64bit) or higher. Platform integration classes are provide for PyQt5, as illustrated in the demo application of the distribution. Please consult the online documentation for further details, in section Topics/Python users.
US 368	Added cee::Vec2d class.
US 367	Remove "Ceetron" folder from start menu path for setup. Windows 8 and 10 do only support two levels in the star menu hierarchy.
US 366	Added a new component, Plot2d, for 2D plotting. Please see the online documentation for more information and example code
US 357	Added support for vector results in local (default) or world coordinates in the UnstructGridModel. If set to world coordinates, the vector directions are not transformed by the part transform matrix. The base position of the vector arrows will of course follow the transformed part. See VectorSettings::setResultInLocalCoordinates() for more info.

1.3.1

US 347	Added support for shader computed flat normals: cee::ug::ModelSettings::setUseShaderComputedFlatNormals(). This settings reduces vertex count 3-6 times (depending on model) compared to the 1.3.0 version. Still a reduction of 50-70% with the new (US345) flat shading optimizations. Note: This technique has some artifacts when combined with ZOOM navigation. When you zoom in with a very large factor the model will start to show a pattern and not smooth colors anymore.This artifact can be avoided by using WALK navigation.
US 346	Added setting of label border width to MarkupPartLabels. Added cee::vis::MarkupPartLabels::borderWidth()/cee::vis::MarkupPartLabels::setBorderWidth().
US 345	Optimized memory usage (CPU and GPU) and rendering speed for part with flat shading (cee::ug::PartSettings::setSmoothShading(false)). Reduced GPU memory usage with around 40%.
US 340	Optimized and added improved support mode-shape animations. Added ModeShapeAnimation class (accessible from the ModelSpec) with settings related to mode-shape animation. See documentation for more info.
US 339	Select parts in the GeometryModel by a polygon in screen coordinates. Added cee::geo::GeometryModel::polygonIntersect(). Extends the regionIntersect() method from a rectangular region to an arbitraty polygon in screen coordinates. See documentation for more information.
US 338	Added support for rendering unlit triangles in the MarkupModel. Added lighting setting to cee::vis::MarkupPartTriangles.
US 337	Select parts in the UnstructGridModel by a polygon in screen coordinates. Added UnstructGridModel::polygonIntersect(). Extends the regionIntersect() method from a rectangular region to an arbitraty polygon in screen coordinates. See documentation for more information.
US 331	Added support for 2D drawing. Added cee::vis::Model::setUse2dPixelProjection(). This makes it possible to render in a pixel exact 2D projection from all models, including both MarkupModel and GeometryModel (and even UnstructGridModel, although that might be a bit far-fetched). Models with this setting enabled will use the x,y part of the cee::Vec3d as (OpenGL based) screen coordinates. The z value of the coordinates will be used to order the items. The one with the largest z coordinate will be visible.
US 325	Report images supports hyperlinks to the full model on Ceetron Cloud
US 318	Create set in the UnstructGridModel by a polygon in screen coordinates. Added DataElementSetGenerator::createFromPolygon(). Extends the createFromRegion() method from a rectangular region to an arbitraty polygon in screen coordinates. See documentation for more information.

1.3.0

US 322	Added a new component for creating reports with interactive 3D models directly from the user application. Please see the online documentation for more information and example code. Please note that the Report component requires a license with this feature enabled. All evaluation licenses have this feature enabled.
US 321	Large performance increase when working with DataElementSet's. The DataElementSetGenerator has been optimized and performs much better, especially on large models/sets. The updateVisualization() method has also been updated when using set filtering (ModelSpec::setVisibleSetIds()).
US 320	Deleting unused data from DataSourceInterface based DataSources in updateVisualization(). This will reduce the memory usage when using file readers and switching between results, states, etc.
US 319	Added setting of the width of the overlay color legend bar (the filled boxes) in the UnstructGrid component: ScalarSettings::setLegendBarWidth()

1.2.1

US 317	Added DataElementSetGenerator class to create DataElementSets based on spacial positioning of the elements in the UnstructGridModel. You have methods to extract sets based on position to a plane and within a region of the screen. Also added Camera::planeFromWindowCoordinates() to create a Plane from a line in screen coordinates.
US 316	Added a flag to DataSourceReader::reload() in order to allow developers to control when the database should be reloaded.
US 315	ImportCAE: Added support for reading 1D elements (cee::ug::Element::POINTS). This applies to all file formats supported by the ImportCAE component.
US 75	Added support for element sets in the UnstructGrid component. Element sets can be used to filter the model by only showing the elements that are in the visible set(s). You can also show the complement of the visible set(s). Added DataElementSet and DataElementSetItem classes. DataElementSets are managed by the DataSource. You specify which sets that should be used for filtering in the ModelSpec with the ModelSpec::setVisibleSetIds() method.

1.2.0

US 313	Updated ImportCAE file readers with new features and bug fixes, which includes: - Support for ANSYS 17
US 312	Added support for user text files in the VTFx file. Any number of text files can be added by specifying a file name and the file content. Use ExportVTFx::addUserTextFile() to write to the VTFx and FileBrowserVTFx::userTextFileCount()/userTextFileName()/userTextFileContent() to access text files in the VTFx.
US 311	Extended the capabilities of SurfacePathQuery to allow computing paths on cutplanes, isosurfaces and isovolumes. Also, querying for element surface result is now supported. Removed path point structure PathPoint in class SurfacePathQuery. Points are now specified using cee::ug::HitItem
US 310	Added triangleIndex() to HitItem. This index corresponds to the index of the triangle delivered by VisualizationPartQuery (normal parts) or CuttingPlaneData, IsosurfaceData or IsovolumeData for cut/iso/isovol. Added triangleIndex as an optional parameter to sampleScalar() and sampleVector() in CuttingPlane, Isosurface and Isovolume to speed up the lookup if you know the triangle index (e.g. after picking).
US 308	Visual Studio 2015 version. From this release we will support and deliver Ceetron 3D Components for VS2015 (32 and 64 bit).
US 307	Extended and replaced ScalarSettings::setAutoFullRange() to ScalarSettings::setAutoRangeMode(). Supported modes are : OFF, ALL_ITEMS, VISIBLE_ITEMS
US 303	Allowed for enabling/disabling surface path sampling. Disabled sampling ensures the path has points on each surface element Added SurfacePathQuery::enableSampling(bool enable) Renamed SurfacePathQuery::sampleScalarResultValues to SurfacePathQuery::getScalarResultValues
US 302	Unstructured grid models support specifying a highlight color. Highlight color is overriden when halo is activated. Added ModelSettings::highlightColor() / ModelSettings::setHighlightColor()

US 301	Ceetron Cloud! It is now possible to upload VTFx files generated by the Export component directly to Ceetron cloud, allowing you to add a Send-To-Cloud button in your app to provide a One-Click-Sharing experience to your users. A topic describing Ceetron Cloud integration has been added to the documentation. Ceetron Cloud integration has been added to the Qt and .NET DemoApp. A SendToCloud button has been added on the toolbar to upload a model to Ceetron Cloud with the click of a button. The source code can be used as a template to integrate OneClickSharing into your application. A standalone command line uploader (sendToCloud) has also been added as a Qt example.
US 299	Introduced a reader for sequential Vtk unstructured grid fils (.vtu) and Paraview data files (.pvd). New DataReaderVtu class in CeeImportCae
US 298	Added atomic reference count in cee::RefCountedObject. Added class cee::AtomicCounter and using this in cee::RefCountedObjects. All addRef/release calls (and thus usage of smart pointers) are now thread safe.
US 296	Added setting of eye lift factor to cee::vis::MarkupPartLabels and cee::vis::MarkupPartText3d. Previous versions used a hard coded value of 2.0, which did not give the desired result. The default is now 0.0 (no eye lift), and to get the old behavior you have to set the eye lift to 2.0 on the MarkupParts.
US 294	Added cee::Vec3f class. cee::Vec3f is mainly by for the VTFx component.
US 293	Added cee::ug::VTFxFileBrowser::readTextFile() to support reading of (custom) text/xml files from a VTFx file.
US 292	Added setting to disable forcing of uniform level heights on the color legend. Useful if using uneven spaced levels and would like the legend to reflect this visually. Added cee::ug::ScalarSettings::forceUniformLevelSpacing.
US 282	Added multi core optimization for several operations, especially in the UnstructGrid component. UpdateVisualization() is now faster, and all the feature extraction (cutting planes, isosurfaces, isovolumes and particle traces) run in parallel using all available cores. Sorting of the rendering queue is also done in parallel, improving the the visualization speed with many parts.
US 269	"Selectability" flag for parts in the GeometryModel. Parts can now be set to not be included in the GeometryModel::rayIntersect() and regionIntersect() methods. This is useful for limiting the parts that can be selected by the user. See PartSettings::setIntersectable().

1.1.1

US 280	Added point size and line width VTFx properties to UnstructGrid component.
US 279	Added controls for ambient and specular intensity in EffectColor, EffectFrontAndBackColor and EffectTexture in the Geometry Component. This allows the user to control the light settings. Reducing the specular intensity and increasing the ambient intensity would lead to less dramatic lighting of the part.
US 276	Added new MarkupModel part: MarkupPartText3d. This part makes it possible to render text that is positioned, oriented and scaled in world coordinates. This allows you to create text that is "glued" onto the model and follow the model when the user performs pan/rotate/zoom. The text 3d layout is described with a position, direction vector, up vector and a text height (all in world coordinates). See cee::vis::MarkupPartText3d in the documentation for more information.
US 275	Added support for showing a special text on the tick marks of the color legend in the UnstructGridModel if all results on the model are out of range of the color legend. See cee::ug::ModelSettings::setColorLegendNoResultOverrideTickMarksText() in the documentation for more information.
US 274	Added option to draw tick mark lines over the legend bar for continuous color legends. See cee::ug::ScalarSettings::setForceTickMarkLinesToCoverLegend() in the documentation for more information.

1.1.0

US 272	Added a new component for reading commercial FEA and CFD files. The Import CAE component provides a DataSource for the UnstructGrid Component that is capable of reading most commercial FEA and CFD files. Please see the online documentation for more information on how to use it and which file formats that are supported. Please note that the ImportCae component requires a license with this feature enabled. All evaluation licenses have this feature enabled.
US 271	Added option to disable use of Vertex Buffer Objects (VBOs) in the UnstructGridModel. This is useful for large animations with changing nodes (displacements or remeshing) and many time steps. See documentation of cee::ug::ModelSettings::setUseVertexBufferObjects() for more information.

US 270	Added support for setting a legend title font in UnstructGridModel. Color legends can now be rendered with a different font for the title and the tick-mark values. Changes to cee::ug::ModelSettings: - Removed colorLegendFont(). - Added colorLegendTitleFont() and colorLegendDetailsFont().

1.0.3

US 267	Added new optimized update action (cee::ug::UnstructGridModel::NODE_POSITIONS) for the UnstructGridModel to use when only the node coordinates have changed (not the number of nodes or the element topology). This will significantly reduce the update time when only modifying the node positions.
US 266	Added support for specifying a font for the OverlayAxisCross and the OverlayNavigationCube classes. The constructors now take a font in addition to a camera. To get the old behavior, modify the constructor to: new cee::vis::OverlayAxisCross(camera, cee::vis::Font::createNormalFont().get()); Added method to set the axis labels for OverlayAxisCross as well as a better layout scheme for the axis labels. The default color of the labels is now white to match other overlay items (e.g. color legend). The default labels are now 'X','Y' and 'Z' (capital letters).
US 262	Added UpdateAction UnstructGridModel::COLOR_LEGEND for use when only the appearance of the color legend has changed (visible on/off, text color, etc). This update is much faster than e.g. UnstructGridModel::SCALAR_SETTINGS.
US 260	Added support for per-frame animation setup in the UnstructGrid component to allow the view to show the frames as they are complete. This is really useful for large models with many time steps that takes some time to setup. The user will see the frames as soon as they are ready and not have to wait until the complete animation sequence is ready. Added the cee::ug::UnstructGridModel::startUpdateVisualizationPerFrame() and cee::ug::UnstructGridModel::updateVisualizationForFrame() methods. See startUpdateVisualizationPerFrame() for more information and an example of how to use this feature. Also added cee::vis::View::requestImmediateRedraw() which redraws the view immediately and does not wait for the normal app message queue processing.

1.0.2

US 258	Unified naming scheme for loggers used by Ceetron 3D Components. See "Topis: Logging" in the documentation for more information.
US 257	Extended clipping support in View to allow multiple UnstructGridModels to use different clipping planes that only affect the model they are defined in. This makes it possible to split a model and show the two parts with different settings (e.g. solid vs. transparent). This is done with two UnstructGridModels sharing the same data source.
US 256	Changed type of fillChar in Str::arg() functions from wchar_t to char
US 255	Added support for custom tick mark values on the color legend when using a CONTINUOUS color mapper. Added ColorMapper::setCustomContinuousTickMarks(). This allows the user to define the tick marks independently of the colors scheme or custom colors.
US 254	Added per frame and per part bounding box query to cee::ug::UnstructGridModel.
US 252	Moved VTFxMemoryFile from CeeExport to CeeUnstructGrid and added cee::ug::VTFxFileBrowser::initialize(ug::VTFxMemoryFile* vtfxMemFile)
US 192	Added support for bounded clipping planes which makes it possible to cut holes in the model with multiple clipping planes and only cut out what is behind a specified number of clipping planes. By setting this number to the number of clipping planes you can e.g. create a box that cuts only what is inside the box and shows everything else. See the documentation of cee::vis::Clipping for illustrations and more information. Also added support for bounded clipping planes for cutting planes in the UnstructGrid model that works the same way as in the cee::vis::Clipping class. See the documentation of cee::ug::CuttingPlane and cee::ug::ModelSettings for more information.