**CEML (Computational Electromagnetic Markup Language):**

***Syntax Specification***

Release Date: 06/08/2012  
Version : 0.4  
Comments : Incremental update – minor changes to implementation.

1. ***Changes from previous version:***

- Boundaries, plane-wave sources, and point sources in the provided examples are now specified relative to the simulation mesh origin coordinates. This change does inherently affect the operation of the CEML standard, but it is recommended that changes be made to reflect this recommendation to any solvers that implement the CEML standard whilst specifying objects relative to a separate origin point.

- Fixed a typo: the tag for plane-wave sources is now spelt correctly. I.e, <planewave></planewave>

1. ***Introduction:***

The motivation behind the development of CEML is to offer a standardised, human-readable method for specifying simulation environments and their contents when using numerical methods such as the Transmission Line Matrix method (TLM) or the Finite Difference Time Domain (FDTD) method. These are two of the most popular algorithms in use for time-domain discretised modelling of CEM-based problems.

Incorporating functionality to parse CEML into their CEM solver applications means researchers and industry professionals can benefit from an open simulation configuration format that allows the exchange of simulation configurations between users. This also means users of different software applications can exchange simulation configurations without need to worry about platform or implementation differences.

1. ***Top-level Simulation Tags***

Every CEML file must first specify a number of ‘top-level’ simulation details that dictate the behaviour of the simulation overall.

* 1. **<simulation>**

The opening statement for any simulation configuration, all subsequent tags (top-level or otherwise) must be contained within the <simulation></simulation> tags.

* 1. **<simtitle> - optional**

This is a user reference title for the simulation.

* 1. **<nodetype> - optional**

An optional tag for selecting a node type if your solver uses more than one numerical method for simulations. Valid values should be all caps in the format of: ‘SCN’, ‘HSCN’, ‘SSCN’, ‘2DSHUNT’, ‘2DSERIES’, etc.

* 1. **<nodesx>**

The number of nodes present in the mesh along the X-axis.

* 1. **<nodesy>**

The number of nodes present in the mesh along the Y-axis.

* 1. **<nodesz>**

The number of nodes present in the mesh along the Z-axis.

* 1. **<timesteps>**

The length of the simulation (i.e. number of time-steps) before the solver is halted.

* 1. **<vismode>**

The default visualisation mode for the simulation. For 2D simulations, valid values are ‘Ez’, ‘Hx’ or ‘Hy’.

1. ***Point Sources***

A point source is an energy emitter that will distribute new energy into the mesh in all directions (dependent on nearby boundaries if present).

* 1. **<pulse>**

The opening statement for a point source pulse object, all pulse-specific configuration tags must be contained within the <pulse></pulse> tags.

* 1. **<xpos>**

The position of the point source pulse on the X-axis.

* 1. **<ypos>**

The position of the point source pulse on the Y-axis.

* 1. **<zpos>**

The position of the point source pulse on the Z-axis.

* 1. **<wavelength>**

The wavelength of the point source pulse (in nodes).

* 1. **<excmode>**

The excitation mode of the point source pulse. Valid values are ‘GAUSS’ for a Gaussian one-off pulse, or ‘SIN’ for a persistent sinusoidal point source pulse of wavelength <wavelength>.

* 1. **<tsstart>**

The starting point for the point source pulse relative to the start of the simulation, in time-steps.

1. ***Plain wave Sources***

A Plain wave source is effectively a one dimensional array of point sources, however the <plainwave> tag acts as a shortcut to creating a plain wave source by simply specifying start and end points for the plain wave instead of every individual point source.

* 1. **<plainwave>**

The opening statement for a plain wave object, all plain wave specific configuration tags must be contained within the <plainwave></plainwave> tags.

* 1. **<xstart>**

The X-axis start coordinate for the plain wave source.

* 1. **<ystart>**

The Y-axis start coordinate for the plain wave source.

* 1. **<zstart>**

The Z-axis start coordinate for the plain wave source.

* 1. **<xend>**

The X-axis end coordinate for the plain wave source.

* 1. **<yend>**

The Y-axis end coordinate for the plain wave source.

* 1. **<zend>**

The Z-axis end coordinate for the plain wave source.

* 1. **<pwtype>**

The amplitude scaling mode along the length of the plain wave. Valid values are ‘STANDARD’ for no amplitude scaling, or ‘WAVEGUIDE’ to scale the amplitude of plain wave output along its length to match one half of a sinusoid for use in a wave guide model.

* 1. **<wavelength>**

The wavelength of the pulse output from the plain wave (in nodes).

* 1. **<tsstart>**

The starting point for the plain wave relative to the start of the simulation, in time-steps.

1. ***Straight-line Boundaries***

Boundaries are useful for represent materials such as metal, or simulating a mesh which allows energy present in it to propagate outwards to infinity. With CEML, in two dimensions boundaries are effectively a one dimensional line, and in three dimensions boundaries are a square sheet.

* 1. **<boundary>**

The opening statement for a boundary definition, all boundary specific configuration tags must be contained within the <boundary></boundary> tags.

* 1. **<xstart>**

The X-axis start coordinate for the boundary.

* 1. **<ystart>**

The Y-axis start coordinate for the boundary.

* 1. **<zstart>**

The Z-axis start coordinate for the boundary.

* 1. **<xend>**

The X-axis end coordinate for the boundary.

* 1. **<yend>**

The Y-axis end coordinate for the boundary.

* 1. **<zend>**

The Z-axis end coordinate for the boundary.

* 1. **<bcond>**

The boundary condition – Valid values are ‘PEC’ for a perfect electric conductor (-1.0 coefficient), ‘PMC’ for a perfect magnetic conductor (+1.0 coefficient) or ‘MTC’ for an absorbing boundary condition that mimics wave propagation to infinity (-((√2-1)/(√2+1)) coefficient).

* 1. **<meshedge>**

Describe the boundary position in the mesh for software that makes distinctions. Valid values for two dimensional simulations are ‘TOP’, ‘BOTTOM’, ‘LEFT’, ‘RIGHT’. Three dimensional simulations can additionally use ‘FRONT’ and ‘BACK’. When implementing an arbitrary internal boundary, use ‘INTERNAL’.

1. ***Circular & Elliptical Boundaries***

Circular boundaries provide an alternative to the straight-line boundaries, and can be used to create representations of more complex objects.

* 1. **<circleboundary>**

The opening statement for a circular boundary definition, all circular boundary configuration tags must be contained within the <circleboundary></circleboundary> tags.

* 1. **<x>**

X-axis coordinate of the centre-point of the circular boundary.

* 1. **<y>**

Y-axis coordinate of the centre-point of the circular boundary.

* 1. **<bcond>**

The boundary condition – Valid values are ‘PEC’ for a perfect electric conductor (-1.0 coefficient), ‘PMC’ for a perfect magnetic conductor (+1.0 coefficient) or ‘MTC’ for an absorbing boundary condition that mimics wave propagation to infinity (-((√2-1)/(√2+1)) coefficient).

* 1. **<ellipseboundary>**

The opening statement for an elliptical boundary definition, all elliptical boundary configuration tags must be contained within the <ellipseboundary></ellipseboundary> tags.

* 1. **<x1>**

X-axis coordinate of the first focus of the elliptical boundary.

* 1. **<y1>**

Y-axis coordinate of the first focus of the elliptical boundary.

* 1. **<x2>**

X-axis coordinate of the second focus of the elliptical boundary.

* 1. **<y2>**

Y-axis coordinate of the second focus of the elliptical boundary.

* 1. **<bcond>**

The boundary condition – Valid values are ‘PEC’ for a perfect electric conductor (-1.0 coefficient), ‘PMC’ for a perfect magnetic conductor (+1.0 coefficient) or ‘MTC’ for an absorbing boundary condition that mimics wave propagation to infinity (-((√2-1)/(√2+1)) coefficient).

1. ***Examples of Usage***

Included in this specification is a sample CEML file for reference, ‘examples.xml’. Whilst these examples are optimised for use with the ‘[Loughborough Wave Lab’ iPhone application](http://itunes.apple.com/gb/app/loughborough-wave-lab/id402424695?mt=8), a TLM-based solver and the first CEM solver to implement CEML compatibility, they should work with any solver implementing the CEML standard.

1. ***Feedback***

Loughborough Wave Lab and CEML are developed by Dan Browne, a Ph.D. student within the School of Electronic, Electrical and Systems Engineering at Loughborough University, UK. If you have any suggestions or other feedback for the improvement and expansion of CEML or the [Loughborough Wave Lab app](http://itunes.apple.com/gb/app/loughborough-wave-lab/id402424695?mt=8), please email: [D.R.Browne@lboro.ac.uk](mailto:D.R.Browne@lboro.ac.uk)