Introduction to Boost.Geometry

presented by Mateusz Loskot (Cadcorp Ltd.)
at FOSS4G 2010
mateusz loskot

- OSGeo charter member, 2007
- GDAL/OGR maintainer 2006-2008
- contributor to OSGeo, GDAL/OGR, libLAS, WKT Raster, GEOS, PostGIS, Feature Data Objects, PROJ.4, libtiff, libgeotiff and others
- with Cadcorp since 2009

http://mateusz.loskot.net
contents

- overview
- design
- features
- performance
overview
what is Boost.Geometry?

• a library dedicated to programmers

• collection of types and algorithms

• solving computational geometry problems

• written in C++ programming language

• header-only library
what is Boost?

full name: Boost C++ Libraries

http://boost.org/

“The Boost C++ Libraries are a collection of free libraries that extend the functionality of C++”

» Wikipedia
history (1)

• 1995 - Geodan Geographic Library

• 2008 - 1\textsuperscript{st} preview for Boost as Geometry Library

• 2009 - 4\textsuperscript{th} preview for Boost as Generic Geometry Library (GGL)

• November 2009 - final review and acceptance to Boost collection as Boost.Geometry
Boost review

• start on November 5, 2009
• review manager: Hartmut Kaiser (Boost.Spirit)
• 14 reviewers
• finish on November 23, 2009
• final report on November 28, 2009
  – 12 votes Yes
  – 2 votes No
  – Several conditions of acceptance
conclusions

“The design is very clear. I think it can serve as a standard example of how to cover a big non trivial problem domain using meta-programming, partial specialization and tag dispatch to make it uniformly accessible by a set of generic algorithms”
future

• incorporate to Boost C++ Libraries
  – work steadily moves on

• release
  – Boost 1.44? or 1.45 or 1.46 or ...
team

• Barend Gehrels at Geodan
  – lead developer and project manager

• Bruno Lalande
  – lead developer

• Mateusz Loskot
  – supporting developer
community

• GGL mailing list
  – http://lists.osgeo.org/mailman/listinfo/ggl
  – ~50 users

• Boost mailing lists
  – http://lists.boost.org
  – very large community with a couple of dozens hackers discussing ideas for Boost.Geometry
users

• Merkaartor (Open Street Map)

• Open Graph Router

• Flight Logbook

• Games (Tangram)

• Geodan
potential
design
challenges

to design and implement a library as

- generic
- fast
- robust
- not specific to any domain
- extensible

programming tool satisfying many with usable
“explosion of capabilities”
technology

• C++ Programming Language
  • ISO/IEC 14882:2003

• C++ Standard Library

• Boost C++ Libraries

• Generic programming techniques
Metaprogramming (generic programming)

template

+ instantiation

+ compiler

= final source code of a specific program
metaprogramming techniques

- **templates** – generic form of source (type)
- **metafunctions** – generate type at compile-time, type selection techniques, encapsulate computation algorithm
- **traits** – associates additional information
- **tag dispatching** – uses traits to distinguish types dispatch calls
- **concepts** – non-intrusive design - “generate” your own library of types and algorithms
- **compile-time strategy** pattern
concepts and models

“A concept is a set of requirements consisting of valid expressions, associated types, invariants, and complexity guarantees.”

“A type that satisfies the requirements is said to model the concept”

- David Abrahams and Jeremy Siek
strategies

template
  +
  type parameters (traits)
  +
  instantiation
  =
  selection of algorithms
agnosticism of dimension

- dimension

point<int, 1, cartesian>

polygon<
    point<double, 2, spherical<radian>>>
agnosticism of coordinate type (1)

- support of different numeric types

\[
\text{point<int, 2, cartesian>}
\]

\[
\text{polygon<}
\]

\[
\text{point<double, 3, spherical<radian>}>}
\]
agnosticism of coordinate type (2)

- support arbitrary precision arithmetic numbers
  - GMP and others (adapted by Boost.Math)
- algorithms select most precise type:
  - int + int → int
  - int + float → float
  - int + GMP → GMP
  - GMP + double → GMP
agnosticism of coordinate space

- points instantiated with coordinate system
- support user-defined coordinate systems
- traits and tag dispatching delegate computations to strategies suitable for specified coordinate system

point<int, 1, cartesian>

definition

polygon<
    point<double, 2, spherical<radian>>>>
features
# Geometry Concepts

## 0-dimensional
- boost::geometry::concept::Point
- boost::geometry::concept::ConstPoint

## 1-dimensional
- boost::geometry::concept::Segment
- boost::geometry::concept::ConstSegment
- boost::geometry::concept::Linestring
- boost::geometry::concept::ConstLinestring

## 2-dimensional
- boost::geometry::concept::Box
- boost::geometry::concept::ConstBox
- boost::geometry::concept::Ring
- boost::geometry::concept::ConstRing
- boost::geometry::concept::Polygon
- boost::geometry::concept::ConstPolygon

## Functions
- boost::geometry::concept::check
- boost::geometry::concept::check_concepts_and_equal_dimensions
### Geometry Models

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-dimensional</td>
<td></td>
<td>boost::geometry::point</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::point_xy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::point_2d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::point_3d</td>
</tr>
<tr>
<td>1-dimensional</td>
<td></td>
<td>boost::geometry::segment</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::segment_2d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::linestring</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::linestring_2d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::linestring_3d</td>
</tr>
<tr>
<td>2-dimensional</td>
<td></td>
<td>boost::geometry::box</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::box_2d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::box_3d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::box_4d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::linear_ring</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::ring_2d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::ring_3d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::polygon</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::polygon_2d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>boost::geometry::polygon_3d</td>
</tr>
</tbody>
</table>

**Adapted:**
Boost.Tuple, Boost.Array, C Array, std::vector, std::deque, std::pair
Core

Metafunctions

boost::geometry::cs_tag
boost::geometry::coordinate_type
boost::geometry::coordinate_system
boost::geometry::dimension
boost::geometry::geometry_id
boost::geometry::interior_type
boost::geometry::is_linear
boost::geometry::is_multi
boost::geometry::is_radian
boost::geometry::point_order
boost::geometry::point_type
boost::geometry::ring_type
boost::geometry::replace_point_type
boost::geometry::reverse_dispatch
boost::geometry::tag
boost::geometry::topological_dimension

Access Functions

boost::geometry::exterior_ring
boost::geometry::get
boost::geometry::get_as_radian
boost::geometry::interior_rings
boost::geometry::num_interior_rings
boost::geometry::num_points
boost::geometry::set
boost::geometry::set_from_radian

Classes

boost::geometry::exception
boost::geometry::centroid_exception
<table>
<thead>
<tr>
<th>Coordinate Systems</th>
<th>Iterators</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Classes</strong></td>
<td><strong>Metafunctions</strong></td>
<td><strong>Functions</strong></td>
</tr>
<tr>
<td>boost::geometry::cs::cartesian</td>
<td>boost::geometry::range_type</td>
<td>boost::geometry::make_segment_iterator</td>
</tr>
<tr>
<td>boost::geometry::cs::geographic</td>
<td></td>
<td>boost::geometry::operator==</td>
</tr>
<tr>
<td>boost::geometry::cs::polar</td>
<td></td>
<td>boost::geometry::operator!=</td>
</tr>
<tr>
<td>boost::geometry::cs::spherical</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Classes</strong></td>
<td><strong>Metafunctions</strong></td>
<td><strong>Functions</strong></td>
</tr>
<tr>
<td>boost::geometry::circular_iterator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::ever_circling_iterator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::one_section_segment_iterator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::section_iterator</td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::segment_iterator</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Algorithms

## Geometry Constructors
- `boost::geometry::make`
- `boost::geometry::make_inverse`
- `boost::geometry::make_zero`

## Predicates
- `boost::geometry::disjoint`
- `boost::geometry::equals`
- `boost::geometry::intersects`
- `boost::geometry::overlaps`
- `boost::geometry::selected`
- `boost::geometry::within`

## Append
- `boost::geometry::append`

## Area
- `boost::geometry::area`

## Assign
- `boost::geometry::assign`
- `boost::geometry::assign_box_corners`
- `boost::geometry::assign_inverse`
- `boost::geometry::assign_point_from_index`
- `boost::geometry::assign_point_to_index`
- `boost::geometry::assign_zero`

## Buffer
- `boost::geometry::buffer`
- `boost::geometry::make_buffer`

## Centroid
- `boost::geometry::centroid`
- `boost::geometry::make_centroid`

## Clear
- `boost::geometry::clear`

## Combine
- `boost::geometry::combine`

## Convert
- `boost::geometry::convert`

## Convex Hull
- `boost::geometry::convex_hull`
- `boost::geometry::convex_hull_inserter`

## Correct
- `boost::geometry::correct`
Distance
boost::geometry::distance

Difference
boost::geometry::difference
boost::geometry::sym_difference

Dissolve
boost::geometry::dissolve

Envelope
boost::geometry::envelope
boost::geometry::make_envelope

for_each
boost::geometry::for_each_point
boost::geometry::for_each_segment

Intersection
boost::geometry::intersection_inserter

Length
boost::geometry::length

Overlay
boost::geometry::copy_segments
boost::geometry::copy_segment_point
boost::geometry::copy_segment_points
boost::geometry::enrich_intersection_points
boost::geometry::get_turns
boost::geometry::traverse

Perimeter
boost::geometry::perimeter

Reverse
boost::geometry::reverse

Section
boost::geometry::get_section
boost::geometry::sectionalize

Simplify
boost::geometry::simplify
boost::geometry::simplify_inserter

Transform
boost::geometry::transform

Union
boost::geometry::union_inserter

Unique
boost::geometry::unique

Miscellaneous Utilities
boost::geometry::parse
<table>
<thead>
<tr>
<th>Strategies</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Area</strong></td>
<td><strong>Buffer</strong></td>
<td><strong>Centroid</strong></td>
<td><strong>Distance</strong></td>
</tr>
<tr>
<td>boost::geometry::strategy::area</td>
<td>boost::geometry::strategy::buffer::join_miter</td>
<td>boost::geometry::strategy::centroid::centroid_average</td>
<td>boost::geometry::strategy::distance::haversine</td>
</tr>
<tr>
<td>boost::geometry::area_result</td>
<td>boost::geometry::strategy::buffer::join_bevel</td>
<td>boost::geometry::strategy::centroid::bashein_detmer</td>
<td>boost::geometry::strategy::distance::projected_point</td>
</tr>
<tr>
<td>boost::geometry::strategy::area::by_triangles</td>
<td>boost::geometry::strategy::buffer::join_round</td>
<td>boost::geometry::strategy::centroid::centroid_average</td>
<td>boost::geometry::strategy::distance::pythagoras</td>
</tr>
<tr>
<td>boost::geometry::strategy::area::hueller</td>
<td></td>
<td></td>
<td>boost::geometry::strategy::distance::cross_track</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Compare</strong></th>
<th><strong>Convex Hull</strong></th>
<th><strong>Distance</strong></th>
<th><strong>Distance</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>boost::geometry::strategy::compare</td>
<td>boost::geometry::strategy::convex_hull</td>
<td>boost::geometry::strategy::distance::close_to_zero</td>
<td>boost::geometry::strategy::distance::haversine</td>
</tr>
<tr>
<td>boost::geometry::strategy::compare::default_strategy</td>
<td>boost::geometry::strategy::convex_hull::graham_andrew</td>
<td>boost::geometry::strategy::distance::fuzzy_equals</td>
<td>boost::geometry::strategy::distance::projected_point</td>
</tr>
<tr>
<td>boost::geometry::strategy::compare::circular_comparator</td>
<td></td>
<td>boost::geometry::strategy::distance::pythagoras</td>
<td>boost::geometry::strategy::distance::cross_track</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>boost::geometry::strategy::distance::haversine</td>
</tr>
<tr>
<td>Intersection</td>
<td>Side</td>
<td>Simplify</td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------------------------------------------</td>
<td>----------------------------</td>
<td></td>
</tr>
<tr>
<td>boost::geometry::de9im</td>
<td>boost::geometry::strategy_intersection</td>
<td>boost::geometry::strategy::simplify::douglas_peucker</td>
<td></td>
</tr>
<tr>
<td>boost::geometry::de9im_segment</td>
<td>boost::geometry::strategy::intersection_points</td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::intersection</td>
<td>boost::geometry::strategy::side::course</td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::intersection::liang_barsky</td>
<td>boost::geometry::strategy::side::side_by_triangle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::intersection::relate_cartesian_segments</td>
<td>boost::geometry::strategy::side::side_by_cross_track</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Boost.Geometry</td>
<td>Side</td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::transform</td>
<td>boost::geometry::strategy::winding</td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::transform::copy_direct</td>
<td>boost::geometry::strategy::crossings_multiply</td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::transform::copy_per_coordinate</td>
<td>boost::geometry::strategy::franklin</td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::transform::degree_radian_w</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::transform::degree_radian_w_3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::transform::from_spherical_2_to_cartesian_3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::transform::from_spherical_3_to_cartesian_3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::transform::from_cartesian_3_to_spherical_2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::transform::from_cartesian_3_to_spherical_3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::transformation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::translate_transformer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::scale_transformer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>boost::geometry::strategy::rotate_transformer</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Policies

#### Compare
- boost::geometry::equal_to
- boost::geometry::greater
- boost::geometry::less

#### Relate
- boost::geometry::policies::relate::direction_type
- boost::geometry::policies::relate::segments_de9im
- boost::geometry::policies::relate::segments_direction
- boost::geometry::policies::relate::segments_intersection_points
- boost::geometry::policies::relate::segments_tupled

### Strategy Concepts

#### Strategy Concepts
- boost::geometry::concept::AreaStrategy
- boost::geometry::concept::CentroidStrategy
- boost::geometry::concept::ConvexHullStrategy
- boost::geometry::concept::PointDistanceStrategy
- boost::geometry::concept::PointSegmentDistanceStrategy
- boost::geometry::concept::SegmentIntersectStrategy
- boost::geometry::concept::SimplifyStrategy
- boost::geometry::concept::WithinStrategy
### Arithmetic

**Add**
- `boost::geometry::add_point`
- `boost::geometry::add_value`

**Subtract**
- `boost::geometry::subtract_point`
- `boost::geometry::subtract_value`

**Multiply**
- `boost::geometry::multiply_point`
- `boost::geometry::multiply_value`

**Divide**
- `boost::geometry::divide_point`
- `boost::geometry::divide_value`

### Products
- `boost::geometry::cross_product`
- `boost::geometry::dot_product`

### Extensions

**TODO**
- ...

---

**Boost.Geometry**

---
performance
“We are aware of the weaknesses of performance tests and that it is hard to design objective benchmarks”

“There are so many differences in behaviour in all libraries under different circumstances, it appeared to be impossible or at least very difficult to compare libraries in one benchmark”

Try it yourself!

http://svn.osgeo.org/osgeo/foss4g/benchmarking/geometry_libraries/
Thank you!

Boost.Geometry
http://trac.osgeo.org/ggl/

presented today with friendly support from Cadcorp