Geosci. Model Dev. Discuss.,
https://doi.org/10.5194/gmd-2017-47-RC2, 2017

# *Interactive comment on* "Shingle 2.0: generalising self-consistent and automated domain discretisation for multi-scale geophysical models" *by* Adam Candy and Julie Pietrzak

**Anonymous Referee #2**

Received and published: 27 June 2017

This paper describes Shingle 2.0 — a Python-based library for the manipulation of spatial domains and unstructured grids for geophysical problems. Through the use of a new XML-based file-format (BRML) and a hierarchy of publicly available software components, Shingle aims to standardise the process of managing the spatial constraints and unstructured grids associated with geophysical domains. To this end, a set of nine "tenets" for geophysical grid-generation are proposed, designed to facilitate the development of consistent and shareable frameworks for unstructured geophysical data and meshes.

The overall idea behind the Shingle library — the development of standardised ap-

Printer-friendly version

Discussion paper

proaches and formats for unstructured geophysical data — is interesting, as current methodologies are clearly ad-hoc. I do however have concerns regarding the distinction between the functionality of the Shingle package itself and the underlying libraries on which it depends. I suggest that a clear summary of the various dependencies be presented early in the paper, with an explicit delineation of functionality. Currently, it appears that:

- The Gmsh package provides the actual meshing capabilities, based on a geometry definition created by Shingle.
- The Spud package is used to support the XML-based BRML file-format. It's Diamond viewer is used for GUI-based file editing.
- Various packages (GDAL, shapely, pyproj) are used to support geometrical operations and queries.
- The pydap package is used for remote data access.

Does Shingle incorporate original algorithms and/or data processing facilities beyond those provided by the underlying libraries? If so, I suggest that these features be documented and novelty demonstrated, etc.

Additionally, I feel that the use of the Gmsh library should not be understated. While Shingle aims to overcome challenges related to the specification of the domain, geometric constraints, etc, I suggest that it is the underlying 'mesh-generation' process that is somewhat more algorithmically and computationally demanding.

Gmsh has also been used for geophysical grid-generation in the past (e.g. Lambrechts et al., 2008: "Multiscale mesh generation on the sphere"), along with a number of other algorithms and libraries, including: Jacobsen et al., 2013: "Parallel algorithms for planar and spherical Delaunay construction with an application to centroidal Voronoi tessellations", Conroy et al., 2012: "ADMESH: An advanced, automatic unstructured mesh generator for shallow water models" and Holleman et al., 2013: (Stomel, in) "Numerical diffusion for flow-aligned unstructured grids with application to estuarine modeling", amongst others. I suggest including a brief review of these previous efforts, demonstrating the benefits of Shingle compared to existing alternatives.

The Candy, 2016 pre-print is referred to throughout, often to provide specific examples of functionality. I suggest that any examples referred to be included in the current paper directly. There appears to be some overlap between these papers, though the Candy, 2016 work appears to focus on more theoretical issues.

Page 2, line 32: ... [is] likely to grow.

Page 4, line 57: "... the meshing process is broken up over multiple parallel threads (as demonstrated in Candy, 2016), ..."

Does Shingle itself manage the parallel meshing process, or is this handled by the Gmsh library?

Page 5, line 102: develop[er]s

---