CS846 Paper Review Form - Winter 2012

Reviewer:
Rafael Olaechea
Paper Title:
An overview of the Scala Programming Language

Author(s):
Odersky, Martin
Altherr, Philippe
Cremet, Vincent
Dragos, Iulian
Dubochet, Gilles
Emir, Burak
Mcdirmid, Sean
Micheloud, St√©phane
Mihaylov, Nikolay
Schinz, Michel
Stenman, Erik
Spoon, Lex
Zenger, Matthias

Nikola
1) Is the paper technically correct?
 [X] Yes
 [ ] Mostly (minor flaws, but mostly solid)
 [ ] No

2) Originality
 [ ] Very good (very novel, trailblazing work)
 [X] Good
 [ ] Marginal (very incremental)
 [ ] Poor (little or nothing that is new)

3) Technical Depth
 [ ] Very good (comparable to best conference papers)
 [X] Good (comparable to typical conference papers)
 [ ] Marginal depth
 [ ] Little or no depth

4) Impact/Significance
 [ ] Very significant
 [X] Significant
 [ ] Marginal significance.
 [ ] Little or no significance.

5) Presentation
 [ ] Very well written
 [X] Generally well written
 [ ] Readable
 [ ] Needs considerable work
 [ ] Unacceptably bad

6) Overall Rating
 [ ] Strong accept (award quality)
 [X] Accept (high quality - would argue for acceptance)
 [ ] Weak Accept (borderline, but lean towards acceptance)
 [ ] Weak Reject (not sure why this paper was published)

7) Summary of the paper's main contribution and rationale
   for your recommendation. (1-2 paragraphs)

The main contributions are introducing the language Scala in a semi-formal way highlighting the mechanisms it provides for abstraction. It describes Scala's the type system, syntax and features comparing it to java and and C#. It shows how scala mixture of novel object-oriented features with functional programming model, plus it's similarity to java, makes it easier to use abstractions in Scala and makes Scala seem at times as flexible as a dynamic language although it is static.

The formalization and overview of the Scala programming language main features and type system can help researchers understand this new language and gain insights about why it has gained such a broad acceptance in a short amount of time. It shows the how the way Scala treats functions as first class objects, allowing inheritance, helps give the language more power and improve the abstraction capabilities. It describes the handling of inheritance, multiple inheritance, object composition and how this help handle abstraction and reuse. It then hints that scale's flexibility make it easier to write good libraries and create constructs similar to those given by a domain-specific language without having to build one from scratch.

8) List 1-3 strengths of the paper. (1-2 sentences each,
identified as S1, S2, S3.)

S1: It gives s succinct description of the Scala type system engine, including it's advanced features (local type inference, covariant types, annotations) and how they improve readability and provide abstraction and

error-checking capabilities.
S2: It relates Scala's new features to other modern languages use of the same feature and gives a rational including for including a feature or not in scala, for instance it explains why usage-site variance annotations of Scala types were discarded from previous versions of Scala.
S3: It shows how class membership and multiple inheritance via mixing helps in building abstractions mechanisms into the language and allows a service oriented component model be incorporated into the language. It shows how this and other features of scala help customize scala to seem like the dynamic language required for a certain domain.

9) List 1-3 weaknesses of the paper (1-2 sentences each, identified as W1, W2, W3.)

W1: There is no empirical evaluation of Scala and how it's used in practice.
W2: It doesn't show the limitations of Scala and it might be too one-sided.
W3: It doesn't discuss in related work languages that have been designed with similar goals to Scala and why they failed.