



Senior Platform Developer

Our platform development group is responsible for making sure that the Talis Platform is the premier environment for developing and delivering great Semantic Web applications. We need your help in designing and building our infrastructure to support hundreds of thousands of users and their data.

We're looking for people who:

- use their code to communicate their ideas clearly
- are proficient in Java and comfortable in Python, PHP and other scripting languages
- can break dependencies and decompose hard problems into simpler ones
- never forget about scalability, performance and security
- prefer to develop test first
- are interested in data modelling with RDF
- can develop solutions to problems, communicate them to the team and get them implemented quickly
- aren't afraid to ask questions
- have implemented HTTP clients and servers
- like to say "let's try it" and "we can do that"
- understand how to balance perfection with reality
- are as happy to lead as to follow
- know when to reuse and when to start afresh
- can tell us about something new they learned this year

How to apply:

Take a look at the problems below, we'd like to hear your thoughts on them. Why not pick two and let us know how you'd go about tackling them? We're not looking for complete solutions, we'd like to start a conversation about it. Please send us your ideas along with a CV to careers@talis.com.

1. The Web can be modelled as a network of nodes labelled with URLs and connected by directed arcs. Suppose we want to find all the URLs linked to and from any given URL, and all the URLs that are linked from any two given URLs. What kind of data structures might be suitable for representing and querying a network with 10^8 nodes each having between 10 and 50 arcs?
2. Discuss the different types of automated testing that are needed to maintain high quality software. What kinds of programming language are best suited to each type of testing? What techniques could be used for testing asynchronous processes and for processes that operate over large volumes of data? Are there any situations that you wouldn't test?
3. Large-scale systems composed of many cooperating application servers often need to share and cache configuration. Suppose any server can initiate changes that need to be reflected in real time to the other application servers in the cluster. What strategies could you use for coordinating this kind of behaviour and how are they tolerant to various failure conditions?