1. What should be the relationship between the consumer and the lambda function? Should that lambda function be a generic service and the applications should know about the lambda function and call it when some event is generated. OR the application should just send a message on an event with event details and lambda service should decide which lambda function to call on that event based on event-function mapping.  
     
   Yes, the latter. The service keeps a mapping of topic to Lambda function, and calls it when that message is received. The point is that the application raising the event should not have to know anything about the Lambdas that might be responding to it.
2. We checked AWS lambda function and how it is implemented and found that Amazon has defined a set of events (which are triggered by other AWS components) and the developer has to create a mapping between the events-source and the function while uploading his code. Will this restrict the scope of the service we are creating as it will not be deployable in multiple clouds? Every cloud will have to define a set of events the developer can choose from. On what criteria do we define these events, if we’re supposed to do that?

unfortunately, I think you’re right, Lambdas won’t be portable across clouds, but I think that’s OK. My preference would be to implement something closest to the AWS Lambda Pull model: (<http://docs.aws.amazon.com/lambda/latest/dg/intro-invocation-modes.html>) and have the user declare the mapping. Services in our cloud would publish their event source details. We would define them incrementally as we enable various services.

1. Should the messaging engine be packaged and reside along with the lambda service OR should be treated as an independent component and reside at a location different from lambda service?  
     
   It should be architected to be separate, the Lambda service would need to be told where its message provider would live. But we may also choose to package a default one.