Asteroid Exploration System
Design Results Overview

Date: 2013-12-20
Author: David Killeffer <rayden7@gmail.com>
Reviewer(s): Jonathan Nichols <jn42887@gmail.com>

Comments from peer design review and optionally the functional review

Jon and I collaborated over email and also over Skype to review each other's designs. John thought my design was very good, and he had a few minor suggestions and corrections he noticed on my initial draft class diagrams. He also suggested that I change the way that Spacrcraft send messages to the Communication Link (my original thought was that I wouldn't even model this because the Spacecraft *objects* would not actually be sending the messages, but rather the *real, physical* Spacecraft robots would be sending those messages. I did change my design to include the Spacecraft class as being able to send these DiscoveryMessages.

What design patterns did you apply in this design?

I used the Singleton pattern for the primary service APIs of each component, as well as the Façade pattern. Since my main objects (Asteroid, Mission, Spacecraft) were both the objects that are being persisted to the database as well as serving as data-transfer objects (DTOs) and they have not too much overlap, I did not think it was appropriate/necessary to establish Factories to handle their creation or set up abstract/interface parent classes for each. In a "real" system, there would almost certainly be several sub-classes for each kind of Mission, Spacecraft, and Asteroid to account for the small (or large) variations within each type.

Did the modular approach to the design help?

Yes, although the delivery of messages from Spacecraft to Mission Management System was certainly problematic, and frankly I don't like my design for this part of the system at all. Another thing that I didn't particularly like was the need to replicate certain enumeration classes in other systems (e.g., replicating MineralType, LifeType, AccessibilityType, WaterType enumerations in the Spacecraft component from the Asteroid component), but that was necessary to try and keep the different components as independent as possible. As I noted in my design document, I think the proper way to design the message interchange in a "real" system would be to use an Enterprise Service Bus (ESB) such as Mule ESB or another one to handle the actual delivery of different types of messages between different components; this would allow for two-way data interchange (so that the Mission

Management System could also send messages that would be received by the Spacecraft, for a variety of purposes).

Did you find that you needed to go back and update modules to support modules designed later?

To a small degree, but not as much as I would have thought. By working on the class diagram and use cases first, and then later working on the interactions between the classes, I did need to go back and update some things (particularly the DiscoveryMessage exchange from Spacecraft to Mission Management System).

Did the design review help improve your design?

It was nice to have another set of eyes to review my design, but the benefits of the peer review at this point were certainly lower than for previous assignments (probably because I'm getting into the habit of writing these documents now!).

Did the reuse of the Authentication Service help or hinder your design?

I think it helped, and having an Authentication Service layered into the design was good because that is a very real-world scenario (such a system would never be made to not be secured and let unauthorized people in). I think the work we did for Assignment 4 laid the groundwork for re-using the Authentication Service in this design, so there is

Do you think you could implement this design?

Yes, I believe I could. I would probably use Hibernate as the ORM layer, and JUnit or another testing framework to handle the automated tests. I would also probably refine this design to use an Enterprise Service Bus (perhaps Mule ESB), although that would significantly add to the complexity of the system, but it would be a much more flexible and richer system.