T. Kelly, A. Kuang, M. Lee, J. Sizoo, M. Swartz

Dr. David Johnson

EECS 448

Sunday, April 24, 2022

Project 4 - Integration Strategy Essay

Integration combines multiple software components into one cohesive system. For project four, our team needed to combine code from different team members and the previous prototype. Our team created a resource management system, with access for students and teachers, including functionality like logging in, creating users, and displaying course information and grades. At the end of project three, the prototype consisted of the interface created using WinForms and the database populating these WinForms using SQLite and C#.

However, at that point the separate tiers for presentation, data, and logic were not linked together to produce an operational product. The database and logic levels needed connection to the graphical interface to allow for user interaction. Therefore, one team member worked on the code linking the database and interface for one of these forms (i.e., the faculty view). Then other team members utilized this codebase as a template, updating as necessary to link the forms for other program views. Additionally, the team members who worked on the user interface had to streamline the design, producing a uniform system. This was accomplished by agreeing on a design standard, and individually updating the different interface forms to comply with this standard.

While there are four different strategies for integration testing, our team utilized a combination of the All-at-Once and Bottom-Up integration strategies. The Bottom-Up strategy integrates by moving from the operational artifacts up to the logic artifacts. Over the project's lifetime we focused first on the user interface in WinForms, which receive messages from the logic and data code artifacts above. However, our team utilized the All-at-Once integration strategy more heavily than Bottom-Up during project four. The All-at-Once strategy runs and tests each code artifact separately, then links them together and tests the final product. This strategy was utilized more heavily because different team members worked on separate software components (i.e., one team member worked on the login page, while another worked on the student view). Therefore, each component was fully fleshed out by a team member before it was linked with the other components. At that point, the final product was tested holistically.