



Inside the Process: *The Architecture Scorecard*

By Justin Gasper, TopCoder Handle: ghostar

June 2008

When a TopCoder architecture competition completes the submission phase, and all the competitors have submitted, the competition enters the review phase. The architecture scorecard is used to determine a winner. The scorecard is used to evaluate numerous facets of an architecture, including core requirements, use of proper technologies, proper scoping and design of individual components, impact on existing systems, and overall quality of the documentation. There are also pieces of the scorecard that address database schemas, testing, and quality of the submitter-provided prototype, which only apply to certain architectures, since the scored items aren't required for all competitions. The TopCoder architecture scorecard has these sections and subsections, with each subsection containing three to four individual questions that are filled in by the review board members.

- Core Application Design (60% of the scorecard points)
 - Functional requirements
 - Major architectural concerns
 - Standards based technical requirements
 - Component design
- Design Details (20% of the scorecard points)
 - Deployment and integration
 - Tiering and logic isolation
 - Data model and entity relationships
 - Design extensibility and flexibility
 - Performance
 - Future scalability of design
 - Assembly
- Documentation (15% of the scorecard points)
 - Written documentation quality
 - Diagrams
 - Testing
 - Component Documentation
 - Secondary Documentation
- Prototype (5% of the scorecard points)

Core Application Design

This section covers the overall approach outlined in the architecture, and whether or not it meets the requirements and will be viable in the long run.

Functional requirements

This subsection is the most important part of the scorecard, making up the largest subsection in terms of points. The four questions in this subsection make up 30% of the overall points awarded. The questions here address that all requirements have been met as laid out in the requirements specification, that each functional requirement has a sequence diagram associated with it, that the algorithms are clearly outlined, and that the architecture being reviewed provides a complete solution for the customer.



Major architectural concerns

This subsection covers major points of the architecture, to ensure that large issues in any architecture are covered, including technology usage, multi-threading and concurrency. This section also makes sure that configuration for the architecture is addressed and will be viable in the long run, as well as any necessary security that may be needed.

Standards based technical requirements

This subsection covers the aspects of the architecture that apply to basic technical requirements, ensuring consistency throughout. This subsection refers to persistence choices, logging, exception handling, transactions in persistence, and internationalization, making sure each of these items has been addressed appropriately in the architecture, inline with the original requirements.

Component design

Component production is a huge part of the software development process at TopCoder, and this subsection ensures that the architecture properly handles components both new and existing. The questions ensure that existing components from the catalog are properly identified and used, cutting down on functionality overlap. The questions also ensure that the new components that come out of the architecture being reviewed are properly scoped and identified. Wherever possible, components should be very generic, ensuring they can be reused in the future, and the scorecard helps the reviewers make this determination.

Design Details

The Design Details section covers individual facets of the submission, ensuring it can be properly deployed, won't negatively affect existing systems, provides all the information related to any database models necessary, and will be properly flexible so that future changes will be easy.

Deployment and integration

To ensure that the architecture will be easily deployed and won't hurt existing systems, this subsection contains questions related to deployment and integration issues. The questions make sure that any changes to existing systems are identified and that the architecture identifies any issues that may arise during deployment.

Tiering and logic isolation

Part of a good architecture for a web or desktop application is ensuring that there is proper delineation between the different layers in the application, including the presentation, business logic and persistence. This subsection contains questions related to ensuring the split between the different layers is well defined.

Data model and entity relationships

A majority of the software developed by TopCoder relies on a database to house the data for the application. This subsection contains questions ensuring the database schema is well defined, will work for the application, and is properly normalized and efficient.

Design extensibility and flexibility

To ensure that the architecture isn't too rigid and will easily allow for future changes and enhancements, this subsection contains questions related to how flexible the design is.



Performance

Performance is a key part of any application, and the performance questions relate to how efficient the design is.

Future scalability of design

Web applications need to be able to scale to larger capacities, depending on usage. This subsection has questions related to how scalable the architecture actually is.

Assembly

This subsection has questions related to how well the architecture will map to an assembly. This helps make sure that the flow of the application through the pipeline all the way through assembly will be easy and when assembly is reached the various component pieces will fit together properly.

Documentation

Submitters in an architecture competition are required to provide documentation in both UML and free text form. The architecture deliverables include class diagrams, component diagrams, and sequence diagrams in UML form, as well as component specifications, and an overall design specification that ties it all together. The questions in the documentation section ensure that all documentation is available, that it is thorough, and that it meets the necessary requirements. This section also covers secondary documentation, like XML schemas, and any additional documentation that may be necessary.

Prototype

Some architecture competitions require a coded prototype to show that the architecture is viable, and to address specific concerns related to how a technology can be used in the actual application being developed. The questions in this section cover the prototype to ensure that it both covers the architecture and can be easily run and deployed for testing.

Overall, the architecture scorecard has questions covering every piece of a submission, ensuring that it meets the requirements, identifies problems and provides solutions to those problems, and that it is properly documented. The scorecard ensures that the winner is the submitter with the very best solution to a given problem.