Applying an analytical framework

Organize and reuse valuable techniques, tools, and examples

Benjamin Lieberman, Ph.D.

April 17, 2007

Modeling complex software and business systems can be a challenging task. Standard design and analysis patterns are helpful but don't usually show you how to apply techniques or demonstrate working examples that are similar to the problem at hand. Fortunately, you can use analytical frameworks to collect and organize analysis patterns, tools, skills, organization techniques, examples, and the expertise of others who have solved similar modeling problems. Learn how to use analytical frameworks to your advantage in various software development areas, from security, messaging and user interfaces, to hiring and training personnel.

Somewhere, sometime, someone has been faced with a problem similar to yours and figured out how to solve it. What if you could capture that person's hard-won information and put the solution in a reusable form? This is exactly what design and analysis patterns have been providing to software engineers for the past decade. However, these patterns generally discuss a specific solution; they don't provide much guidance in learning how to identify the correct pattern and apply that pattern in the first place.

Fortunately, you can turn to analytical frameworks, which combine reusable solutions with analysis patterns, research, useful organization techniques, and specific examples of successful approaches. An analytical framework is a little like your own personal library, tailored specifically to your own experience and background.

Analysis patterns have been used in software development for many years to provide solutions for data access, transaction monitoring and management, security, messaging, user interfaces, and other application needs. They are not meant to be implemented into code, but rather to permit analysts to understand complex problem domains.

Analytical frameworks incorporate such patterns and also provide a checklist of skills, tools, and techniques that are necessary for researching a particular area, such as business analysis or system architecture. This can be a great boon to companies looking for specific skills to meet a current need. If a candidate can demonstrate the listed experience and abilities, it is likely that candidate will be successful with the task.
Understanding the analytical framework structure

Consider for a moment how someone trained in woodworking goes about creating furniture. A worker restricted to a few simple tools and her own personal knowledge may find it difficult and time-consuming to create new pieces. On the other hand, if the same person is supplied with a fully appointed workshop, powerful tools, and a full library of plans and example pieces, she will find it much easier to create complex, interesting furniture. In much the same way, an analyst's performance can be improved by access to a set of proven analysis patterns, powerful analytical tools and techniques, and multiple examples of effective modeling approaches for the capture and presentation of complex information.

An analytical framework is composed of five major components: an assortment of tools, a set of useful solution patterns, one or more model forms, multiple research techniques and skills, and methods for grouping complex information. Table 1 shows the elements necessary for a business analysis framework.

Table 1. Business analysis framework

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tools</td>
<td>Word processing, modeling software, document version control</td>
</tr>
<tr>
<td>Patterns</td>
<td>Industry-specific patterns, business systems patterns, business organization structures</td>
</tr>
<tr>
<td>Model forms</td>
<td>Organization chart, Unified Modeling Language (UML) business case, UML activity, Zachman Framework</td>
</tr>
<tr>
<td>Techniques</td>
<td>Observation, interviewing, document study</td>
</tr>
<tr>
<td>Skills</td>
<td>Note-taking, active listening, meeting facilitation, team leadership, critical thinking, reasoning by analogy</td>
</tr>
<tr>
<td>Categorization</td>
<td>Business process framework, department hierarchy, business use cases, business-functional dependency graph</td>
</tr>
</tbody>
</table>

In this business analysis framework, the effective tools for the discovery and capture of business processes include modeling software (particularly UML modeling, if business requirements are captured as use cases), word-processing aids, and some form of version control for models and documents. Tools can be tailored for specific needs or they can be more general tools, such as a simple visual drawing aid.

Recurring analysis patterns can be effective when applied to a particular analysis domain. In the business analysis framework, these patterns include business system patterns (such as order entry, inventory control, and trade resolution), industry-specific patterns (such as telephony routing, package shipping and scheduling, and drug-manufacturing controls), and business structures (such as matrices, hierarchies, and distributed organizations). Patterns are themes that have been found to occur repeatedly in particular business areas. For more information on ways to capture recurring business themes, refer to "Enterprise Patterns and MDA: Building Better Software with Archetype Patterns and UML." (See the Related topics section for more information.)

You can use a selection of model forms to capture complex information. For complex analytical analysis, a number of useful model forms exist, such as Unified Modeling Language (UML) for
software-intensive systems, Systems Modeling Language (SysML) for system engineering, and the Zachman Framework for organization cross-cutting concerns. In the business analysis framework example shown in Table 1, useful model forms for studying business processes include UML (for process and use-case modeling), hierarchical organization charts, and a potentially modified form of the Zachman model.

Techniques and skills are a crucial aspect of any analysis approach. It is important to be able to provide an assessment of the currently available skills of an individual or team. For example, the technique of interviewing requires the skills of note-taking and active listening. Running a group workshop requires the skills of facilitation and team leadership. System analysis relies upon the abilities to think critically about the problem (recognizing how the problem can be divided into smaller, simpler domains) and apply similar solutions found in other areas.

The goal of analysis is to present the findings to an audience. The best analyst in the world won't be successful if he cannot organize the information in a meaningful, presentable way. Automated tools provide only part of the solution to this problem; the overall organization of the information determines how the intended audience will utilize the data. In the business analysis framework, these categorization principles are represented by business processes (where the business is divided into subdomains), dependency graphs between business use cases, and the overall organization structure (typically, hierarchical).

Applying an analytical framework

Now that you know what comprises an analytical framework, you need to learn how to use one. There are at least three situations where an analytical framework can come in handy. The first is when an individual is required to change roles. For example, a business analyst may move into a system-analyst role or a test-analyst role. Each role has different responsibilities and produces a completely different set of documentation and work products. If the individual can call upon a preexisting analytical framework (with appropriate examples), it will be much easier to prepare for and transition to the new role. The individual must learn which tools, techniques, patterns, models, and organizational approaches work best in each situation. Access to an analytical pattern for each of these areas can greatly reduce the time spent searching for solutions.

The second area where analytical frameworks can prove useful is in assessing the skills and preparation of an outsourcing group. Most organizations that utilize near-shore or offshore outsourcing rely on the past experience of the teams to assess their ability to deliver the agreed artifacts. If the outsource team is held to a particular standard, as defined by a specific analytical pattern that has been proven to be effective, there is higher confidence that the team will have the correct collection of materials and abilities. The same can be said for internal teams, although typically these teams have already proven a particular approach, which provides the source for the analytical framework.

The third area where analytical frameworks come in handy is in training. If you study the framework presented in Table 1, it should be apparent that these can form the basis for a team-improvement training plan. You can perform an assessment to determine whether the proper
collection of framework elements, tools, patterns, and skills are available already within the team members or must be added either through the purchase of a tool or training of the personnel.

Creating and adapting frameworks

An analytical framework is useful only if the tools are familiar to the modeler. Recalling the talented woodworker, the finest tools are useless without the creative application of a trained and talented artisan. Similarly, building aesthetically pleasing models requires adapting one or more analytical frameworks to a problem domain. You need to know how to select the most relevant framework, familiarize yourself with the elements of the framework, and adapt the framework elements to the problem at hand.

Many problems have recurring aspects, so you can often adapt a closely related framework. For example, almost all legacy computer systems need some form of design investigation, if for no other reason than to train new developers in the maintenance and extension of the system. Therefore, a system design framework would be more applicable than a business process framework.

The creation of a new framework may be as simple as cloning an existing framework and adding new patterns and examples. On the other hand, more complex frameworks, such as one that can be developed for the analysis and management of embedded systems projects, may have little in common with other existing frameworks, so you may need to create them from scratch. You should begin by considering the tools and techniques you can apply to the domain. For example, you may need to modify existing techniques, such as interviewing or group facilitation, to support the new domain. You may be able to provide other tools to the modeler for specific tasks, such as computer-aided software engineering (CASE) for organizing content for a federation of Web sites. As time progresses, you can extend the framework to include multiple examples. Finally, at the conclusion of the project, you can formally capture the techniques and patterns that proved most useful as analysis patterns for use on the next applicable problem.

You may need to adapt an analytical framework before it is usable in a particular context. Adaptation of a framework involves the inclusion of new patterns or organizational schemes to better match the needs of the subject. In this way, you can extend and develop these frameworks for novel problem domains. As new tools and techniques become available, and as new analysis patterns are discovered and codified, you can add these to your toolkit and organize them into a modified framework. If the level of modification is significant, it may indicate that you should construct an entirely new framework.

Putting the analytical framework to use

Imagine that you're a senior developer who has just been assigned to a new position of architect for a legacy system. You were uninvolved with the original development work, but you must quickly learn the system and bring a staff of new team members up to speed, which requires you to get up to speed first. Fortunately, you have a few things in your favor. First, you have a good set of example architecture from other similar systems in the company. Second, you have access to the analytical pattern for software architecture (see Table 2).
Table 2. Software architecture analytical framework

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tools</td>
<td>Code profiler, database-reporting tool, software-architecture document template, modeling software, code analysis</td>
</tr>
<tr>
<td>Patterns</td>
<td>Software-deployment patterns, architecture patterns, system-design patterns, data-storage patterns</td>
</tr>
<tr>
<td>Model forms</td>
<td>Dynamic models (such as UML sequence, algorithm charts, UML state, and UML activity), database entity-relationship diagrams, structural models (such as UML class and function call map), data-flow charts, enterprise system models</td>
</tr>
<tr>
<td>Techniques</td>
<td>Abstract thinking, organization and categorization, critical analysis, note-taking</td>
</tr>
<tr>
<td>Skills</td>
<td>Code structure, flow analysis, hardware profiling, system behavior</td>
</tr>
<tr>
<td>Categorization</td>
<td>Functional behavior, subsystem dependency, component dependency, deployment packaging</td>
</tr>
</tbody>
</table>

Using the software architecture framework, your first step is to assemble all of the recommended tools. The modeling tool provides the ability to visualize existing code into a UML model, allowing you to investigate the existing structure of the code. The runtime profiler is valuable for recording call stacks as the application functionality is executed.

Next, you can investigate the common architectural patterns present in the application. You can identify these patterns, if present, by using the common pattern descriptions noted in the framework, especially the patterns of analysis and design. Many such patterns are published; knowing the most common ones will often suffice for an architectural analysis effort.

Familiarity with a variety of model forms, such as entity-relationship diagrams (ERDs), UML, and other system-oriented modeling approaches, will provide you with a wide range of modeling options. Categorizing these models by functional behavior, logical compared with physical structure, and internal behavior states will produce a useful record of the overall architecture for the system. Finally, it is important to know your own limitations. Self-assessment is a difficult task but necessary to ensure and reassure yourself that all of the truly necessary skills are available to the team. Using the set of expected skills listed in the software architecture analytical framework provides a good starting point for determining the kinds of abilities required for an architectural analysis task. The analytical framework provides you with a checklist of the minimum amount of elements you need to be successful. Additional skills may be required for specialized architectures, such as transport networks.

Summary

Studying and understanding complex software systems can be a challenging task. To be successful, you must employ a variety of tools and techniques to gather, arrange, and present system descriptions. By creating and adapting analytical frameworks, you can prepare well ahead of time and collect useful tools and examples to use in your current situation. Now that you know what elements comprise a successful analytical framework, how to reuse frameworks, and how to create new frameworks, you'll be well prepared to tackle many of the common situations encountered on today's complex problems.
Related topics

- For more information on business archetypes, read *Enterprise Patterns and MDA: Building Better Software with Archetype Patterns and UML*, by Jim Arlow and Ila Neustadt (Addison-Wesley Professional, 2003).
- Check out Ben Lieberman's three-part series (PDF) on *The art of modeling*, which teaches you about constructing an analytical framework, model organization and construction, and visual composition (developerWorks, August 2003).
- Learn to model with structural forms, organizing themes, and diagram pivots in Ben Lieberman's article, "*Build effective system models,*" (developerWorks, March 2007).