Learning Management System Evaluation Framework

by Fred M. Beshears 3/27/01

(based on Software Package Evaluation and Selection by Hollander)

---

Major Criteria for Selecting a LMS

- **Known Requirements**
  
  Ability of the package to meet the university's current academic and administrative requirements, and future requirements that are currently known to exist.

- **Unknown Future Requirements**
  
  Ability to modify the package to meet the university's new requirements as they become known.

- **Implementability**
  
  Ability to implement the package easily.

- **Supportability**
  
  Ability of the vendor to support both the package and the University in the future.

- **Cost**
  
  Total cost to purchase and implement the package as well as ongoing maintenance and support costs.
Trade-offs to Consider when Weighting LMS Selection Criteria

There are important trade-offs to consider when assigning weights to these criteria, which should be determined by the University's vision and strategy. For example, a University may want a LMS package that can meet their future requirements and is easy to implement. However, putting an emphasis on meeting future requirements may require a package that uses state-of-the-art component technology (e.g. the Open Knowledge Initiative's open source component framework) even though that technology has not been successfully implemented by other Universities and may contain bugs, making it harder to implement initially.

When a University uses the latest technology and software, it is known as a leading edge or bleeding edge University. If Universities don't understand that it is hard to implement new technology, they understand soon after the start implementation. But, if they believe that having state-of-the-art technology gives them a strategic advantage over their competitors, then the benefits may outweigh the drawbacks of additional effort, cost, and inconvenience.

Other University's want to be close followers. In other words, they want to use a relatively bug-free package, even though they may lose some of the advantage gained by leading edge University's. And, some University's are risk avoiders who may want to drastically minimize their risk by only using well tested and highly reliable software.

Finally, even though a University chooses to be a bleeding edge institution in one area does not mean it has to be so in all areas. For example, some research Universities may want to be bleeding edge in their research, but close followers in teaching and learning.

Relative Weights for the Major LMS Selection Criteria

To help determine the relative importance of the major LMS selection criteria, the evaluators should allocate 100 points among the five criteria, with the most important receiving the highest rating. The following weights can serve as a guide.

<table>
<thead>
<tr>
<th>Criteria Rating</th>
<th>Relative Importance</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>Extremely Important</td>
</tr>
<tr>
<td>35</td>
<td>Very Important</td>
</tr>
<tr>
<td>30</td>
<td>Important</td>
</tr>
<tr>
<td>25</td>
<td>Slightly Above-Average</td>
</tr>
</tbody>
</table>
Then use the criteria rating to weight the raw scores each package receives for that item. The spread sheet could look something like this (the weights listed below are example weights).

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Company Need</th>
<th>Blackboard Raw Score</th>
<th>Blackboard Weighted Score</th>
<th>WebCT Raw Score</th>
<th>WebCT Weighted Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Known Requirements</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unknown Future Requirements</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Implementability</td>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Supportability</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cost</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total Score</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Known Requirements**

When the University develops its current requirements, it should include all requirements that can be anticipated, even though some are not needed now but will be needed in the future. In a corporate planning environment, top management can provide their vision and strategy for the future of the company, which can be used to determine application requirements that are currently know, but not currently in use. In an academic setting, however, instructors who are early adopters of technology can be surveyed to gain some insights into functional requirements other faculty may need in the future. For example, early adoptor faculty may be more willing to experiment with quiz making tools, but other instructors may not until they are convinced the quiz building tools are robust and
easy-to-learn, or if there are other time-efficient ways to implement quizzes (e.g. by selecting quiz items from pre-established quiz item databases).

Also, when developing a list of current requirements, understand that the University will gain little from a new system if the underlying academic processes remain the same. If the only reason to adopt a new system is to be able to perform an inefficient process more efficiently, then the overall cost savings may be incremental at best.

The main question is whether the University should redesign its academic processes before selecting a LMS package or after it is selected so it can use the LMS package to drive the new process. **If the University wants pedagogy to drive the design process, then the University may want to redesign the process before selecting the LMS because it wants the LMS to perform the process in a specific manner.** However, it may be that the underlying academic process cannot be designed in a top-down fashion, so the University may have to buy the LMS package and let its design characteristics drive the redesign of the underlying academic process.

**Known Requirements - User Types**

If the LMS is to be used by multiple user types (e.g. early adoptor faculty, faculty from a well funded department (WFD), faculty from a poorly funded department (PFD), Faculty teaching large courses (TLC), department staff, central support staff), then a particular package may need the needs of certain user types better than those of other user types. To ensure that the LMS selected best meets the University's overall current requirement, determine which user types are most important. This is done by rating each user type in its importance to the University. For example, rate the University's need to support a user type as high (H), medium (M), or low (L), and then convert the H, M, L to a numerical value.

<table>
<thead>
<tr>
<th>Code</th>
<th>Need</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>High (Must serve)</td>
<td>10</td>
</tr>
<tr>
<td>M</td>
<td>Medium (Should serve)</td>
<td>7</td>
</tr>
<tr>
<td>L</td>
<td>Low (Like to serve)</td>
<td>3</td>
</tr>
<tr>
<td>X</td>
<td>No need to serve</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>User Type</th>
<th>Need to Serve</th>
<th>WebCT Percentage Fit</th>
<th>WebCT Weighted Percentage Fit</th>
<th>Blackboard Percentage Fit</th>
<th>Blackboard Weighted Percentage Fit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Early</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Known Requirements - Best of Breed**

When an LMS needs to support multiple user types, there is the possibility of choosing more than one package, or perhaps LMS modules to support different user types, and then integrating them. This is know as *best of breed*. It is a complex and costly solution that may be required to meet important current requirements. However, there is considerable risk involved in completing the integration successfully. **Universities should consider a best of breed solution if their LMS criteria ratings are very high for both Current Requirements and Future Requirements, and very low for Implementability, Cost and Supportability.**

**Known Requirements - Application Requirements**

To determine LMS requirements for managing a type of course (e.g. teaching a large lecture course) one should consider the long-term life cycle of a course website (e.g. initializing the course, assigning long term responsibility for the coursesite, deleting the coursesite), and short-term life cycle of a course website that needs to serve that process (e.g. rostering the coursesite, entering the coursesite in the class schedule, distributing teaching materials, assessing student learning, communicating with students, distributing intermediate results/grades, reporting final results/grades). For each subprocess, define the objective of the subprocess and decompose the process into the steps needed to accomplish the end result.

To aid in this process, there are many websites that list LMS functions of interest to different user types.
Known Requirements - Prioritizing

Once you have identified the current requirements you need to prioritize them to set the target that will determine which vendors should be finalists. You can rank current requirements as follows:

- **H** - High (Must have in order to manage the University's course websites)
- **M** - Medium (Should have to manage the University's course websites)
- **L** - Low (Like to have but not needed to manage the University's course websites)
- **X** - Not needed

<table>
<thead>
<tr>
<th>Ref. No</th>
<th>Requirement</th>
<th>University Need</th>
<th>WebCT Rating</th>
<th>WebCT Cost To Modify</th>
<th>Module</th>
<th>WebCT Weighted Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Easy to use interface</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>124</td>
<td>Sophisticated Quiz Tool</td>
<td>M</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Unknown Future Requirements**

Future requirements do not include requirements that can be anticipated in advance. So, LMS packages cannot be judged in terms of their ability to meet specific future requirements because, by definition, they cannot be known in advance. Instead, you need to evaluate candidate LMS packages according to how easily a package can be made to meet the University's changing requirements. Therefore, future requirements are judged according to the adaptability of an LMS package's conformance to standards and its underlying technical architecture.

Some LMS vendors have made an important commitment to support the standards put forward by the Instructional Management Systems Project. The University may have also adopted standards for portals (e.g. uPortal) and component development (e.g. J2EE), which may affect the evaluation of LMS packages.

Also, the development environment and architecture of the LMS package should be considered when evaluating the other four criteria:

- **Current Requirements**: The architecture impacts the performance of the package as well as its ease of use.
• Implementability: If the pieces of the architecture are not easily integrated or cannot be integrated with related systems such as the online class schedule, there can be trouble implementing the package.

• Supportability: The ability to support the package and its components is directly impacted by the architecture. When a problem arises, the vendor of one piece of the architecture often blames the problem on another vendor, making resolution cumbersome.

• Cost: The total cost of implementing and maintaining the LMS package is affected by the architecture. The cost of software can vary by computer (and whether multiple computers can be used in one virtual, load balanced system), and the size (and number) of the computer(s) and the cost of the peripherals are impacted by the architecture.

Unknown Future Requirements - Architecture

If the University views an enterprise level LMS package as being a major enterprise system (i.e. on a par with student registration), then its architecture will dictate the architecture of the University. The University can either require that the LMS package have a specific architecture to fit the University's requirements, or the University can accept the LMS package's architecture. If the University decided to accept the architecture of one of the LMS packages on the market, the campus should be perform a careful analysis to be sure the package (or packages in the best of breed case) meet the University's requirements.

Unknown Future Requirements - Response Time and Scaleability

A package may meet all of the institutions needs on paper, but cannot respond well when the number of users on the system (e.g. simultaneous quiz tool users) starts to scale up. If LMS vendors cannot provide accurate estimates of the size and number of processors required to support different levels of demand, then the system may fail to meet important future requirements. Some vendors (e.g. WebCT) may provide a mechanism to incrementally add machines/processors and load balancing to support levels of demand that are hard to predict. In general, scalability becomes critical when the number of student users or the level of demand for processor or disk intensive applications changes unpredictably.
Implementability, Supportability, and Cost

Implementability

The ability of a company to take the risk of being a technology leader depends on the company's ability to handle challenging software implementations. The way you rate the implementability for a package depends on where you are and where you want to be on the technology adoption curve. If you want to use technology to gain a strategic advantage, then you need to take more risks with the implementability of the LMS package. However, just because you use technology to gain competitive advantage in some areas (e.g. research) this does not mean you must do so for all your applications.

In other words, a research University may decide to operate some research projects on the bleeding edge of technology, but may decide to be a close follower when it comes to adopting technology to improve teaching and learning. However, if a University decides to be a close follower (or risk avoider) in the area of teaching and learning, then the LMS technology developed at other institutions may end up driving the process of changing underlying teaching practices at the close follower institution.

Implementability Components

Implementability is made up of the following components:

- Vendor background
- Software maturity
- Technology maturity
- Modifications
- Third party implementor considerations
- Implementation assistance from LMS vendor
- Quality
- Documentation
- Training

The first column of the worksheet lists the requirements, and the second column listed the University need rating. How you determine the rating is described below. (The numbers below are just examples. The are not based on any vendor evaluation.)
<table>
<thead>
<tr>
<th>Requirement</th>
<th>University Need</th>
<th>WebCT Rating</th>
<th>WebCT Weighted Rating</th>
<th>Blackboard Rating</th>
<th>Blackboard Weighted Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vendor Responsiveness</td>
<td>10</td>
<td>9</td>
<td>90</td>
<td>9</td>
<td>90</td>
</tr>
<tr>
<td>Vendor Background</td>
<td>7</td>
<td>6</td>
<td>42</td>
<td>7</td>
<td>49</td>
</tr>
<tr>
<td>Software Maturity</td>
<td>3</td>
<td>8</td>
<td>24</td>
<td>8</td>
<td>24</td>
</tr>
<tr>
<td>Technology Maturity</td>
<td>7</td>
<td>7</td>
<td>49</td>
<td>7</td>
<td>49</td>
</tr>
<tr>
<td>Modifications</td>
<td>10</td>
<td>6</td>
<td>60</td>
<td>7</td>
<td>70</td>
</tr>
<tr>
<td>Third Party</td>
<td>10</td>
<td>10</td>
<td>100</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Implementation Assistance</td>
<td>10</td>
<td>9</td>
<td>90</td>
<td>9</td>
<td>90</td>
</tr>
<tr>
<td>Quality</td>
<td>7</td>
<td>8</td>
<td>56</td>
<td>6</td>
<td>42</td>
</tr>
<tr>
<td>Documentation</td>
<td>7</td>
<td>8</td>
<td>56</td>
<td>8</td>
<td>56</td>
</tr>
<tr>
<td>Training</td>
<td>7</td>
<td>8</td>
<td>56</td>
<td>3</td>
<td>21</td>
</tr>
<tr>
<td>Total</td>
<td>78</td>
<td>623</td>
<td>491</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total # of Requirements</td>
<td>10</td>
<td>780</td>
<td>780</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maximum Package Weighted Rating</td>
<td></td>
<td>79%</td>
<td>63%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vendor Implementability Rating</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. **Vendor Responsiveness**

If a school has trouble installing a package and the vendor does not respond in a timely fashion, it will negatively impact the implementation process. But, since
vendors will all claim that they are responsive, you need to ask their clients about their responsiveness.

Vendor responsiveness ratings as they relate to implementability are as follows:
Accept average responsiveness. Rating = 5
Require highly responsive vendor. Rating = 10

2. Vendor Background

Type of Customers - if the vendor has customers similar to you their package may better fit your institution.

Number of Years in Business - the longer the vendor has been in business the likelier the vendor will be able to serve the needs of the University.

Market Share - the bigger the vendor's LMS market share, the more likelier the vendor will continue to be in business.

Ratings:
Vendor background of no importance = 0
Vendor background of average importance = 5
Vendor background of great importance = 10

3. Software Maturity

There are many reasons an LMS package may be hard to implement, including
- the degree of change in the teaching and learning process that may be needed,
- the culture of the institution and its willingness to accept change,
- the quality and knowledge of the staff, and
- the maturity of the software and technology.

If the LMS package has only been beta-tested on a small scale at only a few institutions (i.e. which is the case with both WebCT and Blackboard enterprise level software), then there is a high probability that you will find bugs (or
inability to scale) during implementation. And, the LMS vendor may not be able
to respond quickly if overwhelmed with problems being reported by other
schools.

Ratings for Software Maturity
Willing to accept a new package. Rating = 2
Want a package with average maturity. Rating = 6
Need a very mature package. Rating = 10

4. Hardware Maturity

See software maturity for additional details.

Ratings for Hardware Maturity
Willing to accept new technology. Rating = 2
Want technology with average maturity. Rating = 6
Need a very mature technology. Rating = 10

5. Modifications

The more modifications the LMS package requires to meet the University's
requirements, the greater the risk that there will be problems during
implementation (and with installing subsequent releases of the LMS software).
Some programs can be modified by adding programs to filter data during input or
output to the package. These are know as front-end or back-end modifications
(e.g. modules to enter or extract data in XML format according to IMS
specifications). Other modifications require changes to the code of the package
(know as changes to the code). These changes are more complicated and risk
prone.

Ratings:
Accept heavy modifications. Rating = 0
Willing to accept some changes to the code. Rating = 3
Willing to accept many front-end and back-end modifications. Rating = 6
Willing to accept some front-end and back-end modifications. Rating = 8
Want no modifications to the package. Rating = 10

6. Third party implementor considerations

Some LMS vendors will not assist you in implementing the package or in making modifications. In some cases, a third party implementor can be bigger or more experienced with enterprise applications than the package vendor (e.g. Eduprise).

Rating:
Implementation and modifications to be done with no outside assistance. Rating = 0
Third party of average importance. Rating = 5
Third party very important. Rating = 10

7. Implementation Assistance from LMS Vendor

If you want the vendor to help you implement the package (or if this is required by the vendor as in the case of Blackboard), then you must assess the quality of the vendors support services. This is best done by contacting the vendor's clients.

Rating:
Implement ourselves (no outside assistance). Rating = 0
Accept vendor with below average support. Rating = 2
Accept vendor with average support. Rating = 5
Accept vendor with above average support. Rating = 7
Require vendor with excellent support. Rating = 10

8. Quality
The quality of the vendor's code has an obvious impact on the implementation of the package. If the vendor is under pressure to release new versions of the code quickly, it may contain more bugs than usual and will be harder to implement. If your end users have to contend with bug fixes, they may get discouraged and may even try to persuade other users not to adopt the system. Obviously, quality is very important to all user groups except perhaps the early adoptors. When the quality of a software package is in question, it is important to conduct pilot testing with early adoptors before letting the majority of your end users on the system.

There is an international standard for software development known as ISO9000. Vendors that are ISO9000 certified have passed a rigorous audit by an independent certified auditor. (ISO 9000 certifies the software development process, not the software.)

Rating:
Accept average quality. Rating = 5
Require very high quality. Rating = 9
Require ISO 9000 certification. Rating = 10

9. Documentation

Rate your need for documentation on a scale from 1 to 10, with 1 meaning you have little need for documentation and 10 meaning documentation is very important to you.

10. Training
Training can take different forms:
- Train-the-trainer - Training employees to become trainers, who will then teach other users.
- Computer-based training (CBT)
- Audio and/or video courses
- Texts and workbooks

You can rate your requirements on a worksheet such as the following, with your need for a particular type of training ranked from 0 to 10, with 0 meaning you have no need for this type of training and 10 meaning you must have this type of training.
(The numbers in the table below are just examples.)

<table>
<thead>
<tr>
<th>Training</th>
<th>Need</th>
<th>WebCT</th>
<th>Blackboard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Completeness of training plan.</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Courses meet University's need.</td>
<td>10</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Size of class</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Vendor will customize courses</td>
<td>10</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Course is hands-on</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Computer based training available</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Audio training available</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vidor training available</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Workbooks available</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>70</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>Maximum</td>
<td>70</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Training Rating</td>
<td>83%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Supportability**

Once a package has been implemented, the vendor needs to provide support on an ongoing basis. Also, one reason for selecting a commercial package over a custom-designed solution is that a vendor will continuously enhance the software at a cost that may be lower than custom development (if the University is willing to accept the package without significant modification).

Supportability is made up of the following components:

- Vendor responsiveness
- Quality
- Development methodology
- Modifications
- Financial stability
- Warranty
- User groups
- Support functions

A sample worksheet for rating the supportability of different packages/vendors could be as follows. (The numbers below are just examples.)

<table>
<thead>
<tr>
<th>Criteria</th>
<th>University Need</th>
<th>WebCT Raw Score</th>
<th>WebCT Weighted Score</th>
<th>Blackboard Raw Score</th>
<th>Blackboard Weighted Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vendor Responsiveness</td>
<td>10</td>
<td>9</td>
<td>90</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quality</td>
<td>10</td>
<td>8</td>
<td>80</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Development Methodology</td>
<td>10</td>
<td>10</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Modifications</td>
<td>7</td>
<td>6</td>
<td>42</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Financial Stability</td>
<td>7</td>
<td>8</td>
<td>56</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Warranty</td>
<td>10</td>
<td>8</td>
<td>80</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User groups</td>
<td>7</td>
<td>9</td>
<td>63</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Support Functions</td>
<td>10</td>
<td>10</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>71</td>
<td></td>
<td>611</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maximum</td>
<td>10</td>
<td></td>
<td>710</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. Vendor Responsiveness

To what extent does the vendor keep promises of new features in subsequent releases? Do new releases come out on schedule?

Rating:
Accept average responsiveness. Rating = 5
Require highly responsive vendor. Rating = 10

2. Quality

Does the vendor's help desk provide quality support.

Rating:
Accept average quality. Rating = 5
Require high vendor quality. Rating = 10

3. Development Methodology

The methodology used to develop the package can simplify some modifications. For example, if the package was built with J2EE object oriented technology, you
might be able to modify the behavior of the package by extending the functionality of one or more object classes.

Rating:
Development methodology of no importance. Rating = 0
Development methodology of medium importance. Rating = 5
Development methodology must allow for easy modification of the package. Rating = 10

4. Modifications

If a package needs modifications to support requirements, will the vendor be able to support the modifications in subsequent releases? If not, then whenever the package is upgraded you will have to reapply the modifications (if possible). If the modification is included in the upgrade (e.g. for a fee), then that modification need not be reapplied. Ideally, vendors will incorporate modifications into their standard package so their next release will incorporate your modification.

Rating:
Package is not the type to be modified. Rating = 0
University will reapply modifications to all new releases. Rating = 1
Third party will reapply modifications to all new releases. Rating = 6
Vendor will reapply modifications to all new releases. Rating = 8
Vendor will incorporate modifications into next release of the package. Rating = 10

5. Financial stability

If a vendor is in poor financial condition, it may go bankrupt. Alternatively, if the vendor does not have enough staff to support the help desk, then support will suffer. Or, the vendor may not have the capital or sales revenue to update the package.
Rating:
Vendor's financial condition is of no importance. Rating = 0
Vendor must have a minimum of an average financial condition. Rating = 5
Vendor must have excellent financial stability. Rating = 10

6. Warranty

The vendor should warranty that it will correct any defects that you encounter without charge. How long of a warranty do you want from the vendor (90 days, six months, 1 year, 2 years)? Some vendors provide a minimum warranty for free, and then extend that warranty for an annual fee. On the supportability worksheet, enter the duration you want as a warranty on the package, then rate the importance of the warranty from 1 to 10, with 1 meaning you have little interest in the warranty and 10 meaning that it is very important that the vendor meet your warranty requirements.

7. Users Group and Hub Site

Users groups enable you to share experiences with other users. You can learn how other schools implemented a requirement that the package did not support. They also serve as a forum where you can let the vendor know the types of enhancements your school requires. LMS users groups also exchange "coursepacks" and other online learning materials (e.g. quiz items) through the LMS vendor's hub site.

Rating:
Users groups of no importance. Rating = 0
Users groups are a preference. Rating = 5
National users group is required. Rating = 8
National users group with extensive coursepack offerings via the hub site are required. Rating = 10

8. Support functions

The quality of support provided by a vendor's helpdesk can be very important both for LMS administrators and faculty end users. What type of response is acceptable?
- An engineer who can resolve the problem answers the call.
- A clerk answers the call, records your problem, and then gives it to an engineer who calls you back.
- An answering machine answers and you hope someone calls you back.
- Something in between.

Rating:
Vendor support not important. Rating = 0
Vendor must have guaranteed response time of one hour. Rating = 6
Vendor must have engineer on the Help Desk and available during your work hours. Rating = 8
Vendor must guarantee a resolution time on all products sold. Rating = 10

9. Cost

Generally, the cost of enterprise application packages is 15 percent to 25 percent of the project's total cost. Package implementation is divided into two categories: one-time costs and recurring costs. To develop your cost criteria, allocate 100 points between one-time and ongoing costs (e.g. if you are especially concerned about ongoing costs, then weight that category more heavily).

One-time costs

Each of the one-time cost components should be rated on their importance to you. If the specific cost is of great concern, rate it a 10. If it is of average concern, rate it a 5. If it is of no concern, rate it a 0 (e.g. you don't plan to purchase the item).

Up front costs are:
- software: purchase price including discounts
- hardware: cost of additional hardware required to run and maintain (e.g. test) the software.
- modifications: cost of customizing the package to your specifications
- installation: cost of installing the software and integrating it with your other systems
- conversion: cost of converting data (e.g. course websites on the system that is not chosen)
- training: cost of training the system administrators and those who will become LMS faculty trainers
- additional products: software tools needed to run the system; hardware needed to run the system

Annual Operating Expenses
Each annual operating expense item should be rated based on its importance to you (i.e. 10 - great concern, 5 average concern, 0 no concern). An ongoing cost is of no concern if you don't plan to purchase the item. Typical annual expenses are:
- Annual software license fee
- Annual maintenance base package
- Annual maintenance must have modifications
- Maintenance all modifications
- Maintenance hardware (including depreciation for replacement)
- Maintenance other
- Ongoing University training costs
- Ongoing University help desk costs
- System administrator costs (i.e. for 24/7 support of operating system and hardware)
- Application programmer costs (i.e. for ongoing customizations, installations, and 24/7 support of application software)