Buy / Build
Discussion based on:

Factors affecting the buy vs build decision in large Australian organisations
By Patrick Hung, Graham Cedric Low. 
*Journal of Information Technology.*
London: Jun 2008. Vol. 23, Iss. 2; p. 118
Buy vs. Build

- Strategy & Competitive Advantage
- Cost
- Scale & Complexity
- Maturity / Commoditization
- Time
- Internal resources (staff expertise)
- Risks
- Support structures
Competitive advantage

• Build when the system is for the core processes that differentiate your company.
Cost

• Implementation costs
• Ongoing costs
• Which is likely lower for buy?
• Which is likely lower for build?
"When evaluating whether to buy or build, it's critical to thoroughly understand total costs during the software lifecycle -- typically seven or eight years. This step is important, Lutchen says, because 70 percent of software costs occur after implementation. A rigorous lifecycle analysis that realistically estimates ongoing maintenance by in-house developers often tips the balance in favor of buying."

– Mark Lutchen is the former global CIO of PricewaterhouseCoopers, now head of the firm’s IT Effectiveness practice. (InfoWorld, 2006)
Scale and Complexity

- If simple and easy: build
- If complex: expertise, maturity, and economies of scale come from acquired packages.
Maturity / Commoditization

• "Buy" solutions embody and package "best practices".

• E.g. Consider the feature set of SalesForce. It embodies support for good relationship-management practices.
  – Much more so than you would want to custom-build.

• The maturity of the market will winnow out the poor solutions and refine the best ones.
Advantages of COTS*

• The tool exists and already has many of the functions the organization needs or may need.
• The tool can be tailored to the specific needs of an organization.
• The tool is largely debugged.
• The vendor can provide training, user manuals, and ongoing support.
• The vendor is regularly updating and improving the tool.
• There is often a user community around the product which can be a resource for solving problems.

*Excerpted from Should Nonprofit Agencies Build or Buy a Database? http://www.techsoup.org/learningcenter/databases/page5028.cfm (2/21/07)
Time

• Build will typically take longer.
• Buy allows you to start deployment faster.
  – Unless significant customization is necessary
  – Unless significant work process changes and training is necessary
Internal Resources

• What capabilities do the staff have?
• E.g.
  – Staff is adroit with Microsoft Access
    • Can easily create and support a medium sized database to meet their needs
  – Staff has little technical experience, and no database skills
    • Any solution would need external support.
Risks

• Build allows for managing risks related to the solution not ideally fitting the organization, and allowing for modification.

• Buy puts risks such as development problems and time slips on the vendor.
Support Structures

• Buy allow an organization to rely on professional support structures and/or open source communities
  – Risk: non-responsive or costly

• Build requires building that capacity in-house.
  – Risk: costly
Fully implementing a solution

• What are the major steps to fully implementing a custom-built solution?
## Fully implementing a solution

<table>
<thead>
<tr>
<th>Task</th>
<th>Buy</th>
<th>Build</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define requirements</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Design</td>
<td></td>
<td>√</td>
</tr>
<tr>
<td>Develop</td>
<td>configure</td>
<td>√</td>
</tr>
<tr>
<td>Test / Debug</td>
<td>configuration only</td>
<td>√</td>
</tr>
<tr>
<td>Create staff training material / documentation</td>
<td>adapt</td>
<td>√</td>
</tr>
<tr>
<td>Train staff</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Integrate into daily work processes</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Provide support</td>
<td></td>
<td>√</td>
</tr>
<tr>
<td>Perform regular maintenance (e.g. backup)</td>
<td>√</td>
<td>√</td>
</tr>
</tbody>
</table>
Outsourcing

• Outsourcing is an options if:
  – No COTS solution is viable
  – Building is not a viable solution
    • Not the right staff
    • Solution is too big to learn into
    • Solution requires significant expertise
  – Funds can be allocated
# Outsourcing task

<table>
<thead>
<tr>
<th>Task</th>
<th>Your capacity building task</th>
<th>Outsource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define requirements</td>
<td>√</td>
<td>reduced</td>
</tr>
<tr>
<td>Design</td>
<td></td>
<td>√</td>
</tr>
<tr>
<td>Develop</td>
<td></td>
<td>√</td>
</tr>
<tr>
<td>Test / Debug</td>
<td>Test use</td>
<td>√</td>
</tr>
<tr>
<td>Create staff training material / documentation</td>
<td>joint</td>
<td>reduced</td>
</tr>
<tr>
<td>Train staff</td>
<td>partial</td>
<td>reduced</td>
</tr>
<tr>
<td>Integrate into daily work processes</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>Provide support</td>
<td></td>
<td>√</td>
</tr>
<tr>
<td>Perform regular maintenance (e.g. backup)</td>
<td>plan</td>
<td></td>
</tr>
</tbody>
</table>
Deciding on a solution

• Create a simple decision matrix
  – One axis: requirements
  – Second axis: products

• If decision is non-obvious
  – Assign weight to each requirement
  – Assign score to how each product meets each requirement
  – Sum the products
  – See if the outcome “feels right”