The Problem
When your web service changes, the clients consuming the service might stop working. This forces your customers to rewrite the client and to reconsider the cost of doing business with you. Not all changes need to break clients. With some planning, your services can be made flexible enough for future changes, without jeopardizing your customers’ applications. However, without a clear best practice, this is case-specific.

Our Solution
To work towards a best-practice for managing web service change, we implemented a simple web service, changed it, and documented the results for clients using .NET and Java EE.

WSDLs provide two mechanisms for adding flexibility for future changes. Setting minOccurs=0 allows elements to be missing. This provides a mechanism for removing data from messages. Secondly, ‘any’ allows services to add arbitrary data to messages.

<table>
<thead>
<tr>
<th>Change Made to the WSDL</th>
<th>Effect on Client</th>
<th>Client Status / Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition of Data</td>
<td>Client cannot interpret the additional data</td>
<td>.NET client will ignore additional data, Java Client will break</td>
</tr>
<tr>
<td>Removal of Data</td>
<td>Client may require the missing data</td>
<td>Specify minOccurs zero, mark as deprecated</td>
</tr>
<tr>
<td>Addition of a Web Method</td>
<td>Client cannot call new operation</td>
<td>Client will not break</td>
</tr>
<tr>
<td>Removal of a Web Method</td>
<td>Client may require the missing operation</td>
<td>No Solution, mark as deprecated</td>
</tr>
<tr>
<td>Alter Data Type</td>
<td>Type mismatch causes misinterpretation</td>
<td>Add additional variable instead of replacing old one, mark old one as deprecated</td>
</tr>
</tbody>
</table>

The following 3 examples are all legal messages to send according to the above WSDL segment:

1. `<location>`
   - `<street>123 High St.</street>`
   - `<city>Columbus</city>`
   - `<state>OH</state>`
   - `<location>`
2. `<location>`
   - `<street>123 High St.</street>`
   - `<city>Columbus</city>`
   - `<state>OH</state>`
   - `<zip>96382</zip>`
   - `<location>`
3. `<location>`
   - `<latitude>38.3838</latitude>`
   - `<longitude>-40.3331</longitude>`
   - `<geocode>`
   - `<location>`

Developing web services has become a preferred paradigm for developing applications with reusable components which can be accessed by a variety of systems regardless of their platforms over a network such as the internet.

While web services provide flexible, reusable components across a variety of platforms, additional care must be taken when modifications to the web service must be made. Since web services and their clients communicate according to a contract, the service’s WSDL, poorly planned modifications to the service can lead to customers no longer being able to use the service, or worse breaking all together. By following the best practices of web service versioning, developers may produce web services which can be modified without risk of failing to provide service to clients who may not be using the newest contract. This leads to an overall more successful service and more satisfied customers.