```
   </current_location>
  </user>
 </users>
</users_getInfo_response>
```

Thrift generates similar code for declaring RPC function calls, serializing into known output structures, and turning internal exceptions into external error codes. Other toolsets such as XML-RPC and SOAP provide some of these benefits as well, perhaps with a greater CPU and bandwidth cost.

Employing a beautiful tool like Thrift provides recurring benefits:

Automatic type synchronization
: Adding `'favorite_records'` to the `user` type or turning `uid` into an i64 needs to happen across all methods consuming or generating these types.

Automatic binding generation
: All the messy work of reading and writing types is gone, and translating function calls into XML-generating RPC methods requires the function declaration, type checking, and error handling that Thrift does automatically.

Automatic documentation
: Thrift generates a public XML Schema Document, which serves as unambiguous documentation to the outside world, usually much better than what one finds in "the manual." This document can also be used directly by some external tools to generate bindings on the client side.

Cross-language synchronization
: This service can be consumed externally by both XML and JSON clients, and internally over a socket by daemons in all types of languages (PHP, Java, C++, Python, Ruby, C#, etc.). This requires metadata-based code generation so the service designer isn't spending her time updating each of these with every small change.

We now have the data component of a social web service. Next we'll figure out how to establish these session keys to enforce the privacy model users expect on any extension of Facebook.

A Simple Web Service Authentication Handshake

A simple authentication scheme makes this data accessible within the Facebook user's notion of privacy. A user has a certain view into the data of the Facebook system, based on who the user is, his privacy settings, and the privacy settings of those connected to him. Users may authorize individual applications to inherit this view. What is externally visible to a user through an application is a significant portion of (but no more than) what the user could see on the Facebook site itself.

In the architecture for a separate application site (Figure 6-1), user authentication often takes the form of cookies sent from a browser, originally assigned to the user after a verifying action