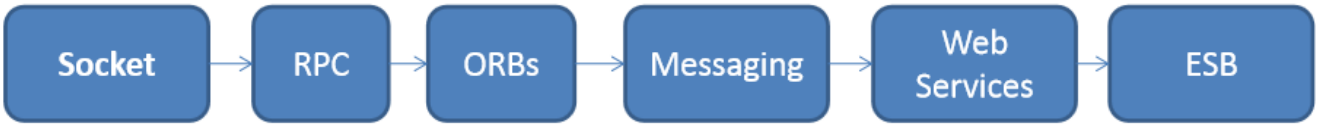
Following figure shows the six classical Software Integration methods.



Development methods of Information System Components

Below we present the basic methods by which IT system components can be developed. IT systems and some of their components differ from traditional stand-alone applications in that, **we expect them to function as a service with almost constant availability**.

However, in order for an application/software to function as a service, it immediately raises the following questions:

- How to control the life cycle of the component?
- How can we manage the resources of the component?
- Where/how can we get the configuration information needed to run?
- How can we communicate with the environment, or the other sw components?

Native development method

Although this is the oldest method, but it is still used in many ways today. For example, in embedded systems, or in containers.

We compile the source code for a specific CPU and Operating System combination: e.g. (x64/Ubuntu). It is possible to fine-tune the code to be run, to optimize its speed or size. The requirement to run the components continuously as a service, requires advanced software developer knowledge.

- c/c++/d compilers: msvc, gcc, clang, dlang
- strong knowledge of: pointers, references handling, heap/stack memory management, multi-threaded resource management
- it is the responsibility of the developer to release the allocated memory can be a big challenge: API calls must be known at the operating system level
- the integration of the various components is difficult: custom serialization methods must be implemented