CURRENT SOFTWARE STACK LIMITATIONS
Diamond has a well-established stack of applications developed over many years which provides users with Data Acquisition and Analysis functionality from the Controls Hardware interface right up to the live and offline post processing and visualisation of experimental data. However the stack has the following limitations:

• Controls, Acquisition and Analysis layers tend to have been developed by different teams, in some cases leading to hard boundaries of technological and operational knowledge.
• Generic Data Acquisition (GDA) software has grown up organically over a period of 15 years leading to bad structure and other forms of technical debt making it hard to maintain and difficult to develop.

To support Diamond II, we want to review our software with a view to designing a revised platform architecture to address these and other problems and to take advantage of industry best practises and technologies. Our goal is to:

• Repackage the existing proven functionality in a more flexible structure behind a common platform API.
• Revise some existing implementations, adding new capabilities and features along the way.
• Migrate to a stable consistent platform that is easier to maintain and support.
• Move toward a solution that offers more flexibility to cross the old boundaries to get to the information required by the user.

API
Due to our requirements of discoverability and reactivity ideally with WebSockets compatibility there are relatively few established technologies to choose from. Facebook’s GraphQL is a strong candidate though which has obviously been tested and debugged at scale in various languages and is well supported.

Gateway and Security Controller
Requirements for API hosting, Authentication and Authorisation are ubiquitous in Web based applications which we can take advantage of. This is true across several languages and many can provide Single Sign On support with little developer effort. Because most user interaction will take place at the Acquisition and Analysis level, a Java based solution is likely and Spring Cloud is a very strong contender in this area. It meets these requirements and is very well supported and documented. Despite this we will look for other contenders in both Java and Python.

Messaging Backbone
Many options are available here, Active MQ is already in use for similar purposes in Diamond and offers strong routing functionality. Also under consideration is Kafka which gives the ability to replay past messages in the event of a failure and is optimised for streaming which may be important at the Data Analysis level for post processing tasks. Both these and others are again well supported and debugged.

Reactivity
Both Python and Java have ReactiveX implementations that offer the required functionality.