SOFTWARE ENGINEERING METHODS

1. The company needs a predictive model. In this regard the Waterfall model, V-model and Sashimi model are options to consider. The requirements for the software are known and no changes are required.

   "… the requirements from the client perspective are very well known and do not need to change".

   Again the model would not require validation and verification since the requirements are fixed. Hence there is a perfect transition process from requirement to implementation. The team also has experience building the software. However, the team is dispersed so the model to be adopted must be fairly easy to implement. In effect, a **Waterfall model** would be the best option for their use case in this scenario.

2. The Waterfall Model

   The Waterfall model (illustrated in **Fig.1**) implements each step in the Software Development Cycle in a sequential manner. The current phase is completed before moving to the next phase.

   It follows through sequentially right from the requirement stage to the maintenance stage. In this regard the requirements for the software should be well known and fixed throughout the phases.

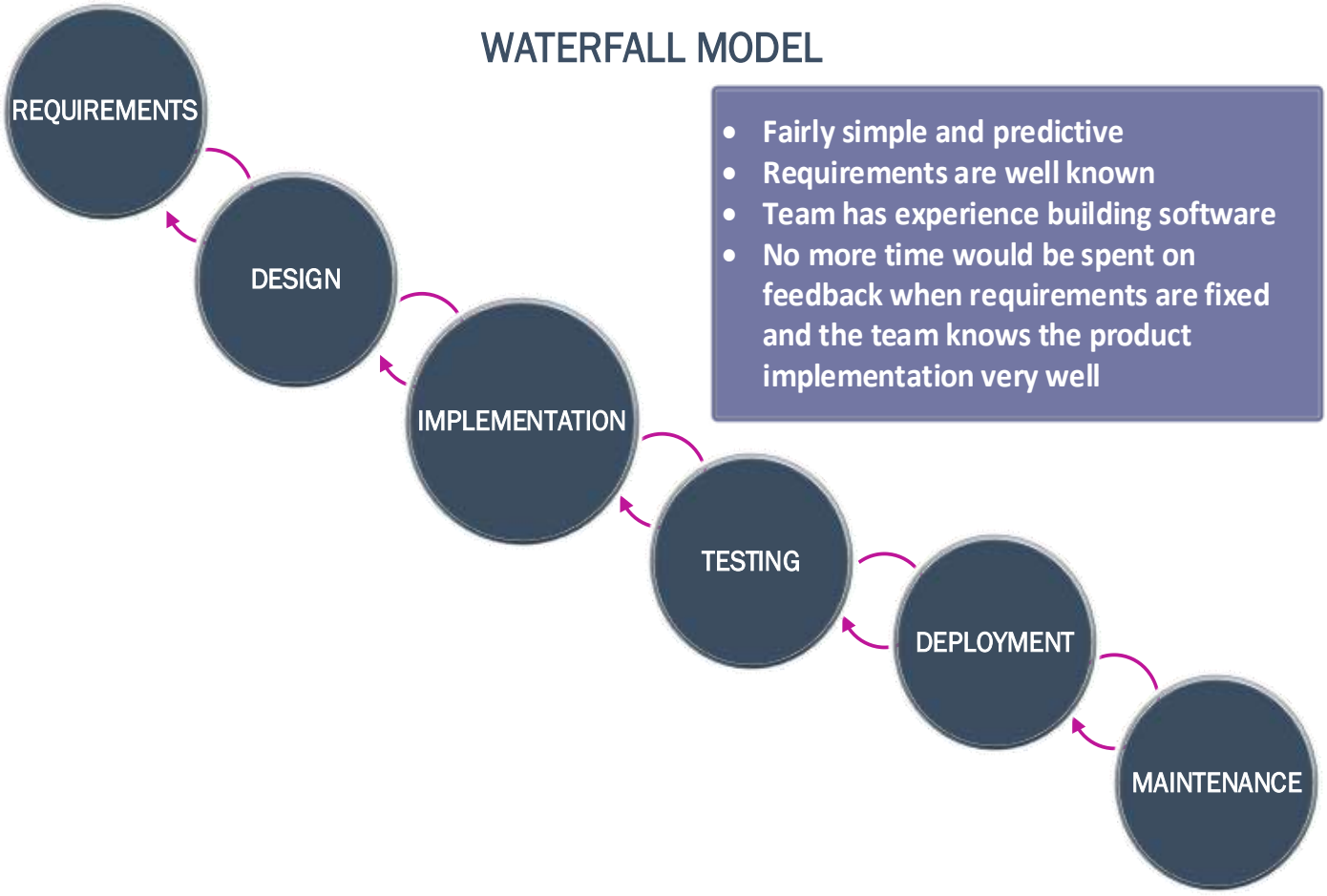

Fig.1. Process Flow in the Waterfall Model

3. The team ultimately seeks scalability. In this regard system testing should be held in high esteem. However, majority of system testing is basically handled at the various stages of testing. Hence I recommend the product to go through all the testing phases. The team should first of all plan the test and develop various testing strategies, take into consideration monitoring and analysis as well before the design stage. The team should consider as many as possible test suites available together with their respective test cases. During the test implementation, the team should consider every live environment associated with the software and execute the tests meticulously.

   As discussed earlier, the team of testers should diligently perform the system testing. However prior to system testing the team of developers and testers should perform the unit and integration tests respectively. After the system test the stakeholders and users should be encouraged to perform the acceptance test (both alpha and beta) despite the fact that the users already love the product. Black-box testing should not be left out and is to be conducted.

4. There should be a test suite for registration associated with employees. Some of the test cases would include valid and invalid employee personal information check.

   There should be a test suite for login of employees. This would also include valid and invalid entry of employee credentials.

   There should also be a test suite for daily operations. The portal would contain records including inventory keeping, payroll, etc. Test cases should be written for each field.

   There should be a patient portal test suite which would test cases associated with patients' medical records, appointment scheduling, etc.

   There should also be a test suite related to billing and payments. All the various tools and processes associated with billing and payment should have related test cases.

   Finally, medical records, including patient history should be protected and kept confidential. Hence there should be a test suite associated with standards related to medical record security. The individual test cases should well written.