**Team Si (Silicon)**

Benjamin Olson

Jorge Alvarez

Anna Waldron

Victor Shaw

1. **Waterfall with a risk reduction model (Spiral into Waterfall)** 
   1. The project should use a spiral into waterfall to develop the Red Planet Explorer. Since the Red Planet Explorer failed, the ultimate goal is to ensure the explorer does not explode and damage NASA’s reputation. A spiral into waterfall addresses the requirement, architectural, and other risks associated with the project. The science team needs to address the “quality and ability sensors” of the probe and conduct other experiments too. All of these potential risks are addressed at the start of the project. Using the spiral into waterfall will allow you to develop a prototype, test experiments, and gather requirements.
   2. The NASA project uses subcontractors to build and design various parts of the explorer. The spiral into waterfall’s system integration and testing phrase combines individually developed components into one complete system. This ensures the system is validated and verified the completed system with the original requirements.
2. **Waterfall with subprojects** 
   1. Waterfall with subprojects is the best option for this project because of all the individual sections of the website. A waterfall with subprojects model allows the project to move forward after finishing certain sections. Delaying the implementation of certain parts of the website due to the slow development of more difficult sections is inefficient.
   2. There is also a predefined user interface template, scripts from the original system, and requirements. On top of that, no change is forecasted in the development, so there would be no need to backtrack up the process, making a waterfall with subprojects life cycle the most optimal option.
3. **Scrum or Code and Fix**
   1. Code and fix is the best option for developing the Rainbow Fish website. The development process is being done by one person, so many aspects of SDLCs are not necessary for coordination and ensuring that the pieces all fit together. The requirements are laid out ahead of time, though there may be adjustments as there is no contract and the overall customer developer relationship is very laid back.

1. **Unified Process**
   1. Unified Process is the best model to use in this instance. What is required is a system that encompasses and centralizes computing and use of a medical equipment network. There are no future expectations or requirements that will be changed further down the development process. The equipment is stated to monitor hospital patients through the use of a system that can be accessed remotely perhaps using a type of server or private network, accumulating a lot of risks. Unified process addresses any issues with concern to risk using UP phases, primarily addressed in the inception phase and then double-checked in the elaboration phase.
   2. Since the risk is high due to the goal of monitoring patients remotely, a prescriptive model would be most appropriate to maintain control throughout the process by following a structured and well-formatted plan for high assurance. This lends itself to using a more cleanroom development model to avoid defects by using formal methods such as precise models and rigorous inspection since people's lives will depend on this system.
   3. Due to these reasons, I would say that a Spiral Model could work but more so a Unified Process model as both are risk-driven involving incremental progress. However UP is a better fit because it can be used with a large or small team, and as it is not specified it is a good fit and has an emphasis on continuous quality verification.
2. **Evolutionary Prototype**
   1. Evolutionary Prototyping is the best model for this instance because there is high uncertainty with requirements or goals with a smaller group of developers but allows the company to work without high risk or solid specifications. This also works well because of the visible steady signs of progress necessary for a funded startup. Since the goal is vague and about improving life and the universe this suggests user feedback as it seems tailored towards the user, demonstrating the evolution of the goal to be determined by user feedback and tool usage.
3. **Extreme Programing**
   1. Extreme programming is the best option for developing the cloud-based applications for Cloud Technology. Extreme programming is usually done in small teams of 2 to 12, and the team in this problem is 7. There also is a need to develop the software quickly for the client, which is typical of XP. XP is an agile SDLC, which will allow the development team to take the input from the team member participating in the standards committee and acts as a marketing representative.
4. **Spiral into Waterfall**
   1. Spiral into Waterfall is the best model for this instance as the specifications and requirements are already detailed out and at the forefront since it is a contract. The Spiral into Waterfall model allows for some flexibility since it is a separate company working for another. This model uses prototypes to better understand what is expected, with a tangible basis of development which would keep the team more on track with the due date. The model also addresses the problem of delivering a product that does not meet requirements, which is explicitly defined in the contract, almost as a form of risk assurance.