Problem Sheet 2. Model selection for software development

Your goal for the practical is to select an appropriate process model for a particular case scenario. When performing that, think about the problem in terms of its novelty to the world (e.g., try to understand whether this is a well-known problem or a completely new one) and to your company, analyze the availability of information on user needs; try evaluating your experience in the area, expected project risks, the need for incremental deployment, expectations from customer management and their willingness to collaborate, and other involved factors.

Problem 1. Briefly explain what risks or risk classes are *highly* likely to occur to the following projects. Use the provided Risk Assessment Sample Checklist if you like.

1) The Federal Aviation Administration (FAA) wants you to rewrite the U.S. air traffic control system.
2) A small sports supply store wants you to write some sales software, but the two partners can't agree on exactly what it should do.
3) Your customer wants you and your team of five fearless developers to write a word processing application as powerful as Microsoft Word in the next three months. (At double your normal rates!)
4) A real estate developer wants to build an application that helps design large housing developments. (Your team just finished building a vacation costing application.)
5) Your customer dumped a 10-page software specification on your desk and then left on a 3-week vacation.
6) Your customer wants to build a 3-D printing application that lets you make buildings out of concrete. (Really, search the Internet for "3-D printer castle Andrey Rudenko" to see what one bored contractor did with this idea in his spare time.)

Solution to Problem 1.

1) Project with long schedules;
   costs might increase from year to year;
   subcontractors as part of the development effort;
   extremely large number of potential stakeholders;
   extremely large size of the project.
2) Contradicting/ambiguous customer requirements;
   changing customer requirements.
3) The project is significantly larger or smaller than what the team is used to working on;
   large size of the project;
   unrealistic schedule estimates/expectations.
4) The development team is unfamiliar with the environment being used;
   The project is significantly larger or smaller than what the team is used to working on.
5) Risks to the project caused by requirements that are inadequately defined.

6) High risk relating to new unreliable technology;
   The development team is unfamiliar with the environment being used;

Problem 2. Indicate which of the following tasks would be better handled predictively or adaptively and briefly say why.

1) Manufacturing parts of cars
2) Building a pedestrian bridge over a river
3) Following a series of clues in a scavenger hunt (aka a quest)
4) Making a scavenger hunt (aka a quest) for others to follow
5) Planning a picnic
6) Planning a picnic in Saint Petersburg
7) Planning a major motion picture
8) Teaching an introductory programming course
9) Finding a specific restaurant while visiting an unfamiliar city before 1990
10) Finding a specific restaurant while visiting an unfamiliar city with a GPS
11) Finding a specific restaurant while visiting an unfamiliar city in a few years when cars drive themselves and are plugged into a smart street network

Solution to Problem 2. The following list explains whether the tasks would be better handled predictively or adaptively.

1) This is a distilled example of a well-understood task with extensively known and widely available technology that may furthermore nearly be ordered online in 2020. Cost and schedule savings should be top priority for such types of projects, not adaptive execution.
2) This is a well-understood task, so you can do it predictively. (Unless you're doing something weird like trying an experimental architectural technique or building an art project.)
3) Because you don't know where the clues will lead you, you're going to have to be adaptive. You might not even know how many clues there will be.
4) When you follow a scavenger hunt, you don't know where each clue will lead. When you're building a scavenger hunt, however, you have control over the clues, so you can handle it predictively.
5) This is a mostly well-understood task, so you can do it predictively. There may be some uncertainty about the weather, so you should use risk management to have a backup plan in place.
6) In Saint Petersburg, there's less question about the weather: It will be wet. You can still do this predictively, but now the dry weather plan should probably be the backup plan instead of the primary plan.

7) A major motion picture is a huge undertaking involving hundreds of people and months or even years of work. Sometimes things go wrong during shooting, but the basic schedule is more or less fixed, so this is a predictive task.
8) This is mostly predictive because you need to hit certain milestones within a set period of time. For example, in a semester you need to cover a certain number of chapters, giving a reasonable number of tests, and regularly hold office hours. Those tasks all fit nicely into a predictive model. At the same time, a class's focus often wanders around a bit depending on the students' interests, how well they can sit still for certain subjects, and the instructor's creativity. Students learn best if the subject is something that interests them personally. For example, you might ask students to build a simple database application but let them pick the domain. They could store car specifications, football team statistics, dessert recipes, or information about their DVD collections. This requires the instructor to be adaptive. To handle both the predictive and adaptive needs, the instructors I know use predictive lesson plans that explain more or less what will happen during class with varying levels of detail depending on the instructor. Then they adjust the lesson in progress and sometimes modify future lessons if necessary.
9) Without GPS you would look up where the restaurant is on a map (or call it and get directions), plan out a route, and follow it. If something went wrong (like a DeLorean breaking down in front of you), you would be more or less stuck. You could reroute (if you didn't leave the map at the hotel), but it would take significant time and effort.
10) This would still be predictive, only this time the GPS would plan the route. If something went wrong (like an SUV running out of gas and blocking the road in front of you), you could make the GPS plan a new route. However, the GPS unit's user interface can make it cumbersome to plan a new route that avoids a particular blockage.
11) This would probably still feel predictive to you because people like to know what's going to happen. (This is one of the more attractive features of predictive models, particularly for management and customers.) However, the car and its computer systems could handle this completely adaptively. The car would plot a route and present it to you for your piece of mind. Then as the drive progresses, the car could watch for blockages (like a flying saucer landing in front of you and blocking the road) and instantly revise the route if necessary. That means this would probably look predictive to you, and you would follow the original plan most of the time, but behind the scenes the car could handle this adaptively.

Problem 3. Briefly explain why each of the following projects might be risky if approached with a predictive process model.

1) The Federal Aviation Administration (FAA) wants you to rewrite the U.S. air traffic control system.
2) A small sports supply store wants you to write some sales software, but the two partners can't agree on exactly what it should do.

3) Your customer wants you and your team of five fearless developers to write a word processing application as powerful as Microsoft Word in the next three months. (At double your normal rates!)
4) A real estate developer wants to build an application that helps design large housing developments. (Your team just finished building a vacation costing application.)
5) Your customer dumped a 10-page software specification on your desk and then left on a 3-week vacation.
6) Your customer wants to build a 3-D printing application that lets you make buildings out of concrete. (Really, search the Internet for "3-D printer castle Andrey Rudenko" to see what one bored contractor did with this idea in his spare time.)

Solution to Problem 3. All these projects have trouble indicators for predictive projects.

1. This is an incredibly large and complex project. A predictive project would provide a level of control that the FAA would probably like, but the sheer size of the project would make it difficult.
2. Lack of clear vision. If you do pry requirements out of the partners, they'll probably be inconsistent and unclear.
3. Unrealistic expectations and lack of resources. It might be possible to build this application predictively, but probably not in 15 person-months.
4. Lack of experience. Your team has experience with vacation-costing not housing development. (Unless you're not telling me about a previous project.)
5. Lack of user involvement. Unless the customer is willing to help define the requirements, it will be hard to satisfy the specification you've been given. (Although you could spend the 3 weeks working on requirements and then finish them up after the customer returns from vacation. That would probably work, all else being equal, but it doesn't seem like a promising start to a new project.)
6. Unestablished technology. Perhaps 3-D concrete printers will be common in a year or two, but until then, this would be very speculative. (However, it would also be very fun! I'd be tempted to take the project anyway, although perhaps with a different development model.)

Problem 4. Under what circumstances would a predictive model cost less in time and effort than an adaptive model? Under what circumstances would it cost more?

Solution to Problem 4. A predictive model can save money if you correctly plot out the development effort's path. Adaptive models sometimes take extra time chasing unprofitable lines of development and refactoring.
However, if you don't map out the development plan correctly and need to make major changes, a predictive project can cost much more than an adaptive approach.

Problem 5. Suppose your customer wants an application with 10 features and insists that the application is completely useless unless all 10 are implemented with full fidelity. Would there be any benefit to iterative, incremental, or agile approaches?

Solution to Problem 5. The iterative, incremental, and agile approaches enable you to release partial applications as soon as you've implemented enough features with enough fidelity to be useful. However, you don't have to take advantage of that capability if you don't want to. You could still use those approaches to build the application and give the customer the application only when it was complete. Then you would still get the other benefits of those techniques. For example, those approaches would help you refine the requirements throughout the project.

Problem 6. Look at the Unified Process scheme. Why might the deployment tasks start during the elaboration phase instead of at the beginning of the transition phase? What deployment tasks might you be performing during elaboration? What tasks might you be performing during construction?

Solution to Problem 6. The deployment tasks include everything necessary for deployment. You can start working on some of those tasks as soon as you know what's necessary. If you wait until the start of the transition phase, you'll probably be late because you can't purchase and install things such as desks and computers instantly.
During elaboration, the requirements start to coalesce, so you can start writing user documentation. You can also start planning the physical setup (items such as desks, chairs, computers, printers, and networks). However, you shouldn't commit to those items too early in case plans change later. (It would be expensive to buy 1,000 computers early in the project only to discover later that you need a different type of computer or that you need only 150 of them.) So you can start planning the installation of the physical equipment during elaboration, but you generally won't actually start installing that equipment until the later iterations of construction when plans are more concrete.

Problem 7. Look at the Unified Process scheme. The testing tasks begin in the inception phase before the implementation tasks start. What are you testing during inception if there isn't any code yet?
Solution to Problem 7. During the inception phase, the team can test the project's general ideas, assumptions, and approach. Team members can think of scenarios that need to be handled and decide whether the requirements can handle them. Those thought experiments can help refine the project's overall shape.

Problem 8. Look at the project shown in Figure 13-6. What kinds of code are the team members writing during the elaboration phase? What kinds of tests are they performing during that phase?

Solution to Problem 8. Code written during the elaboration phase is usually exploratory. It is written to try out new techniques and ideas that may be used in the application's design. The results of those experiments help refine the requirements.
You don't need to do a lot of testing on exploratory code because that code won't be used in the final application. Tests performed during elaboration are more likely to be applied to the requirements and design. For example, you can create and walk through use cases to verify that the application's design can handle them.

Problem 9. Suppose you're a real estate developer building a neighborhood containing 100 houses. How would each of the predictive, iterative, incremental, and agile approaches correspond to home sales? Assume the "features" of the project are the houses and "releasing a feature" means allowing people to move into a home. Which of the approaches could work? Which approach do developers actually use?

Solution to Problem 9. In a predictive model, you wouldn't let anyone move in until every house was complete. That would work.
In an iterative approach, people would immediately move into every house but with limited fidelity. Here "limited fidelity" would mean the houses initially wouldn't have water, sewer, electricity, appliances, or (worst of all) cable TV and Internet access! This wouldn't work very well. (Although it seems to have been the approach the Russians used when they built the athlete housing for the 2012 Olympics.)
In an incremental approach, people would move into each house as soon as it was completely finished. Some people would move in relatively early and some would move in later. This would work well.
In an agile approach, people would move into each house as soon as it was partly finished with low fidelity. Over time, the occupied houses would be improved and new houses would be started. This wouldn't work. (Unless, perhaps, you're building a refugee camp.)
In practice, developers want to get money out of the project as soon as possible, so it's natural to try for an agile approach. Unfortunately, people can't move into houses that aren't finished. Not only would people refuse (at least I would), but it's a lot easier to install carpeting, paint walls, and finish hardwood floors before people move all their junk into a house. However, it's not too hard to do the landscaping while people live in the house.
So here's the typical approach. First, specialized teams move through the development. First, one team pours the foundations. After the foundations are dry, another team does the framing. Electricians and plumbers move through next, and so on with other teams installing drywall, roofing, carpeting, appliances, and everything else in waves. Sometimes, the teams can even work at the same time. For example, one team might be installing plumbing on one house while another installs drywall on another house.

Problem 10. Your company is a leader in providing solutions related to installation of software systems for human resources (HR). Your company is being hired by a medium-size retailer to provide this solution. Which process model is suitable for this use-case?

Solution to Problem 10.

- The case represents a very well-known problem, that has been previously approached multiple times and with a known solution.
- Your company is an expert and can predict risks and plan ahead easily.
- Therefore, strongly predictive Waterfall models such as vanilla Waterfall, sashimi, or V-model can be easily used.

Problem 11. A very large distributed hospital wants to automate their processes across its divisions (think automatic accounting for acquiring medications, patient health records, etc.) and has hired your company that has substantial expertise in similar automation projects, except not at this scale. The hospitals' management is unsure about nuances that may arise during such an automation. A few hospital divisions will benefit from immediate automation. Which process model is suitable for this use-case?

Solution to Problem 11.

- The case represents a problem that is in principle well-known.
- The solution to the problem is also well-known, as automation systems have been existing for a while now, except that the scale of the solution may be unique.
- The customer will likely benefit from phased delivery, indicating the need for a gradual rollout of the future system.
- Nevertheless, all divisions should be supplied with the same system.
- All this indicates that a subtle variation of a Waterfall model may be useful.
- Recommended an incremental Waterfall model with requirements and design phases shared across all increments.

Problem 12. The defence agency wants to build a novel capability to keep the country protected from potential conflicts. The system has never been attempted, including nearly zero literature existing on the topic. The system is expected to be significantly big and complex, taking decades to build. The agency and your company both have only vague ideas about the possible solutions. A huge number of stakeholders get involved in building the project. Which process model is suitable for this use-case?

Solution to Problem 12. This project may be characterised by:

- Unknown customer needs and project outcomes.
- Very high levels of risk related to customer requirements, design approaches, implementation, etc.
- The system is intended to be very large and complex.

- All this indicates an increased need to perform extensive risk mitigation when performing the project, suggesting a variant of a spiral model focusing on risk.
- Additionally, iterative risk reduction on each increment may be suggested