Aidan Polivka

Discussion Board 2

Part 1

I think the term “models” is highly overloaded in systems and software engineering. We could be talking about designing and modelling software, or we could be talking about project development models. For this discussion, I’ll try to talk briefly about both.

- What are the strengths and weaknesses of models?

  Each common project development model has it’s own strengths and weaknesses. Depending on the chosen model, pivoting and changes to the system design can be difficult to account for. For example, the waterfall method (which consists of major pre-work design) does not offer a lot of wiggle room for change. Flexibility is a weakness for waterfall, but a strong suit for iterative project design strategies like Agile. Agile has problems with staffing and client participation. Both the client and the engineers on the project have to be committed to the development style, and a lot of times Agile project require a lot of staff to maintain the constant iterative design strategy.

  Software development designs are another story altogether. Documented designs like system flow charts and entity relationship diagrams are typically inflexible and go out of date quickly. Depending on the design strategy, they can also be difficult to change. An example of this would be mermaid diagrams. A lot of times it’s better to document software in detail AFTER major features or snapshots have been developed, and use shallower, less detailed models for initial design. The major benefit of software design modelling is documentation so that developers stepping into a project can be quickly onboarded. It can also help with long term documentation for the developers taking over maintenance of the system. Finally, software development models help with bringing context to all engineers working on a feature.

- What impact does modeling have on systems engineering?

  Project development models can make or break your deliverable. Moving into a development project without a plan can result in a loss of client trust and can be detrimental to velocity. Also, starting a project without all stakeholders, project managers, and engineers buying into the system engineering model can cause problems during development of the project.

On the flip side, clearly defining the system engineering model and having dedicated members on the project create a well oiled and flowing project. This in central to having good project and development velocity, while maintaining client trust.

Software design models can help support the engineers in their efforts to maintain the context of the project in development. It can also be a useful tool in explaining to product owners and stakeholders what the prioritized work is supposed to accomplish. Finally, good documentation and development models can help with maintenance after the handoff from contractors to the long-term support engineers.

- How are models used in systems engineering to understand system performance, capabilities, and limitations?

  There are additional models and metrics that can be compiled over the course of a project that allow for better understanding of the health of a project. For example, task burndown charts are a commonly used metric to show how much work has been completed versus how much work was committed to throughout a sprint. These metrics can also help team leads and project managers understand how much work can be committed to, and it can help tech leads estimate how much time it'll take to complete a given task or user story.

  For software development, flow charts and event modelling can help developers understand time complexity of operations and system load. Entity relationship diagrams can help the database analyst and engineers understand the business rules and limitations of their current software architecture, and support tech leads estimations on the difficulty of implementing new features.

- How are models used to reduce development costs?

  A development road map created in the conceptual and preliminary design stages can drastically reduce development cost. Understanding the path of the project, upcoming work and required client decisions before enlisting more engineers and staff to a project can cut costs for the client. This means that in the initial stages of design, only a few individuals need to be assigned to the project. Also, having a well-defined high level view of the project will reduce the number of questions and pivots mid-development.

  Software design models mostly save engineers time. Whether it's for on-boarding, tracking down a bug, or adding a new feature to existing software, having documentation and diagrams that reflect the existing system will save engineers a lot of time researching.

Advantages and Disadvantages of various Software Models. (2021, March 22). GeeksforGeeks. https://www.geeksforgeeks.org/advantages-and-disadvantages-of-various-software-models/